

Realtime  
publishers

"Leading the Conversation"

*The Shortcut Guide<sup>™</sup> To*



# Managing Disk Fragmentation

*sponsored by*



*Mike Danseglio*

## Introduction to Realtimepublishers

by Don Jones, Series Editor

For several years, now, Realtime has produced dozens and dozens of high-quality books that just happen to be delivered in electronic format—at no cost to you, the reader. We’ve made this unique publishing model work through the generous support and cooperation of our sponsors, who agree to bear each book’s production expenses for the benefit of our readers.

Although we’ve always offered our publications to you for free, don’t think for a moment that quality is anything less than our top priority. My job is to make sure that our books are as good as—and in most cases better than—any printed book that would cost you \$40 or more. Our electronic publishing model offers several advantages over printed books: You receive chapters literally as fast as our authors produce them (hence the “realtime” aspect of our model), and we can update chapters to reflect the latest changes in technology.

I want to point out that our books are by no means paid advertisements or white papers. We’re an independent publishing company, and an important aspect of my job is to make sure that our authors are free to voice their expertise and opinions without reservation or restriction. We maintain complete editorial control of our publications, and I’m proud that we’ve produced so many quality books over the past years.

I want to extend an invitation to visit us at <http://nexus.realtimepublishers.com>, especially if you’ve received this publication from a friend or colleague. We have a wide variety of additional books on a range of topics, and you’re sure to find something that’s of interest to you—and it won’t cost you a thing. We hope you’ll continue to come to Realtime for your educational needs far into the future.

Until then, enjoy.

Don Jones

Introduction to Realtimepublishers.....	i
Chapter 1: Introduction to Disk Architecture .....	1
Introduction to Disk Architecture .....	2
Hard Disks and Disk Architectures.....	4
Disk Interfaces .....	5
IDE.....	5
SCSI .....	6
SATA .....	9
Disk Interface Wrap Up.....	9
Fault-Tolerant Disk Systems.....	10
RAID 0.....	10
RAID 1 .....	11
RAID 5.....	11
RAID Wrap Up.....	11
File Systems .....	12
FAT .....	12
FAT12.....	13
FAT16.....	13
FAT32.....	13
NTFS.....	13
Other PC File Systems .....	14
HPFS .....	14
ext3.....	14
File System Wrap Up.....	15
How Disks Are Used .....	15
Fragmentation .....	16
Disk Bandwidth .....	17
Summary .....	17
Download Additional eBooks from Realtime Nexus! .....	17
Glossary .....	18

## Copyright Statement

© 2006 Realtimepublishers.com, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtimepublishers.com, Inc. (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtimepublishers.com, Inc or its web site sponsors. In no event shall Realtimepublishers.com, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtimepublishers.com and the Realtimepublishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtimepublishers.com, please contact us via e-mail at [info@realtimepublishers.com](mailto:info@realtimepublishers.com).

[**Editor's Note:** This eBook was downloaded from Realtime Nexus—The Digital Library. All leading technology guides from Realtimedpublishers can be found at <http://nexus.realtimedpublishers.com>.]

## Chapter 1: Introduction to Disk Architecture

Computers were initially designed to store and process data. Little has changed in this regard since the invention of the modern computer in the mid-20<sup>th</sup> century. However, the scale has increased tremendously. Computers process an immense amount of data, and that data must be stored somewhere. On modern computers, that storage is usually a hard disk.

Storing data on a disk has become less expensive and more convenient in modern times. Hard disk prices are, at the time of this writing, incredibly inexpensive. 750GB of hard disk storage, which just 5 years ago required a large disk array and cost tens of thousands of dollars to plan and implement, costs less than \$400 for a single off-the-shelf drive unit.

But with that increase in storage capacity and decrease in price comes a problem of management. Unfortunately, most modern administrators are complacent about their hard disks. Little care is taken to ensure that the disks continue to perform at their best based on numerous recent Gartner Group surveys. But with very little work, these disks can be maintained in optimal condition and provide exceptional performance for years.

This guide will explore the storage of data on hard disks. This chapter will examine how modern computers and operating systems (OSs) implement disk storage and access. Later chapters will explore one pervasive problem with such storage—fragmentation. This guide will also examine how fragmentation affects a computer system and what approaches are effective in reducing the effects. This guide is comprised of four chapters:

- Chapter 1: Introduction to Disk Architecture—Explains disk structure and storage techniques at a basic level. This information is essential to understanding the problems of modern data storage, including disk fragmentation.
- Chapter 2: Issues with Disk Fragmentation—Describes why disk fragmentation is a problem. Although performance is usually cited as the only fragmentation-related problem, there are a number of symptoms that can result from a fragmented disk.
- Chapter 3: Solving Disk Fragmentation Issues—Explores the most effective methods to both prevent and remediate disk fragmentation issues. Each remedy is analyzed to help you decide which is best for your environment.
- Chapter 4: The Business Case for Defragmentation—Analyzes the decision-making parameters around fragmentation. Cost/benefit analysis is performed from several viewpoints to help you make a decision about how to solve this problem.

## Introduction to Disk Architecture

To understand disk performance, it is necessary to take a brief look at disk architecture, which is usually a misunderstood topic as it can be technically complex and has a number of variables that change as technology develops. You should have a basic understanding of disk operations to help you make the right choices in your disk management strategy. Thus, this guide will introduce you to the most basic and universal concepts that will help you understand the business problem and possible solutions. This guide is not intended to be a compendious reference to disk architecture.

The disk is connected to the computer through several layers of connectivity that make up the data pathway. The disk itself has its own controller and storage architecture. For the computer to understand the disk storage, a common interface must be defined. This interface is the disk interface. For the OS to communicate with the disk interface (and hence the disk architecture), some type of intermediate system must be in place. This is the file system. Each of these elements is discussed in this section.

Figure 1.1 illustrates most of the components in the data pathway between an application, such as Microsoft Word, and the actual disk subsystem that stores and retrieves its data. The graphic is slightly simplified to omit some of the less-important parts of the data pathway. The blue arrows indicate transitions from one transmission media to another and are common disk throughput *bottlenecks* (described later).

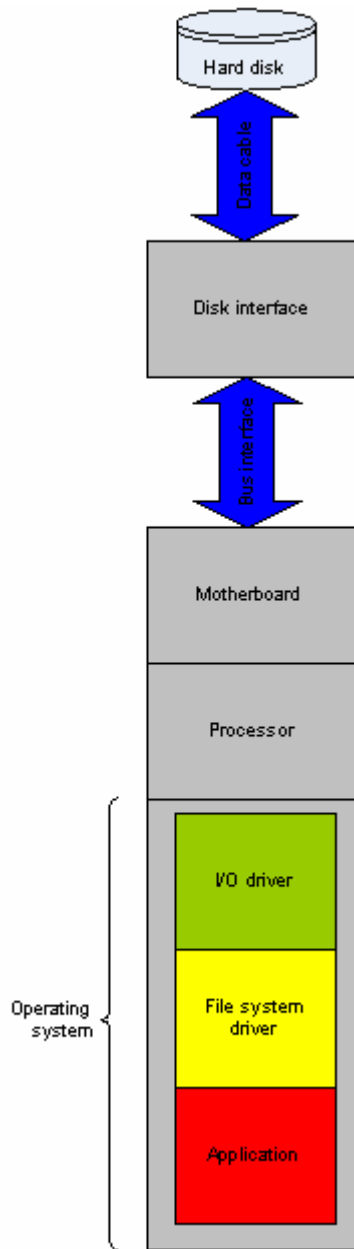


Figure 1.1: The data pathway between an application and a hard disk.

## Hard Disks and Disk Architectures

Hard disks have evolved greatly over the past few years and are very different from the earliest examples in 1955. At their core, they are simply persistent data storage devices. Data can be stored on them and retrieved later. For the purposes of this guide, you need to know only a few things about the hard disk itself.

A hard disk has one or more platters, which are circular surfaces that store data. The data is written to and read from the surfaces by a read/write head, or simply head. The platters spin very quickly while the head moves over them closely, reading and writing the data as the platters spin. The head actually applies a magnetic field as it moves across the disk, which is a smooth magnetic surface. The data is stored as 0s and 1s corresponding to whether the point on the disk is magnetized or not (see Figure 1.2).



**Figure 1.2:** A hard disk spinning while the head accesses the data (Source: <http://www.flickr.com/photos/alphasix/179676660>).

The speed of the spinning platters, the head, and the interface all contribute to the speed of the disk. For that reason, disks are often listed with those speeds as major selling points. For example, disks with platters that spin at 10,000RPM are priced higher than disks with the same storage capacity that spin only at 7200RPM or 5400RPM. The higher RPM usually corresponds to faster disk throughput. (Unfortunately, it also often corresponds to more heat generation and power consumption.)



There is one common misnomer that should be made clear to avoid future confusion. Disks store data in units called sectors. One sector is the smallest part of a disk that can be discretely addressed. When you occasionally see an error such as “Sector not found,” the error is referring to that portion of the physical disk.

Clusters are logical groups of one or more sectors. Clusters are normally defined by the file system, which is implemented by the OS. They help optimize the use of disks by abstracting applications from the physical disk geometry. The abstraction allows the application developer to concentrate on simpler read and write applications without having to know detailed information about disk subsystems, and it allows the OS more complete and efficient control over the disks.

## Disk Interfaces

The connection between the hard disk and the motherboard is called the disk interface. The data coming from the computer’s processor to the disk must be converted to a transmission method that the disk can understand, and the same transformation must be made when data goes from the disk to the motherboard.

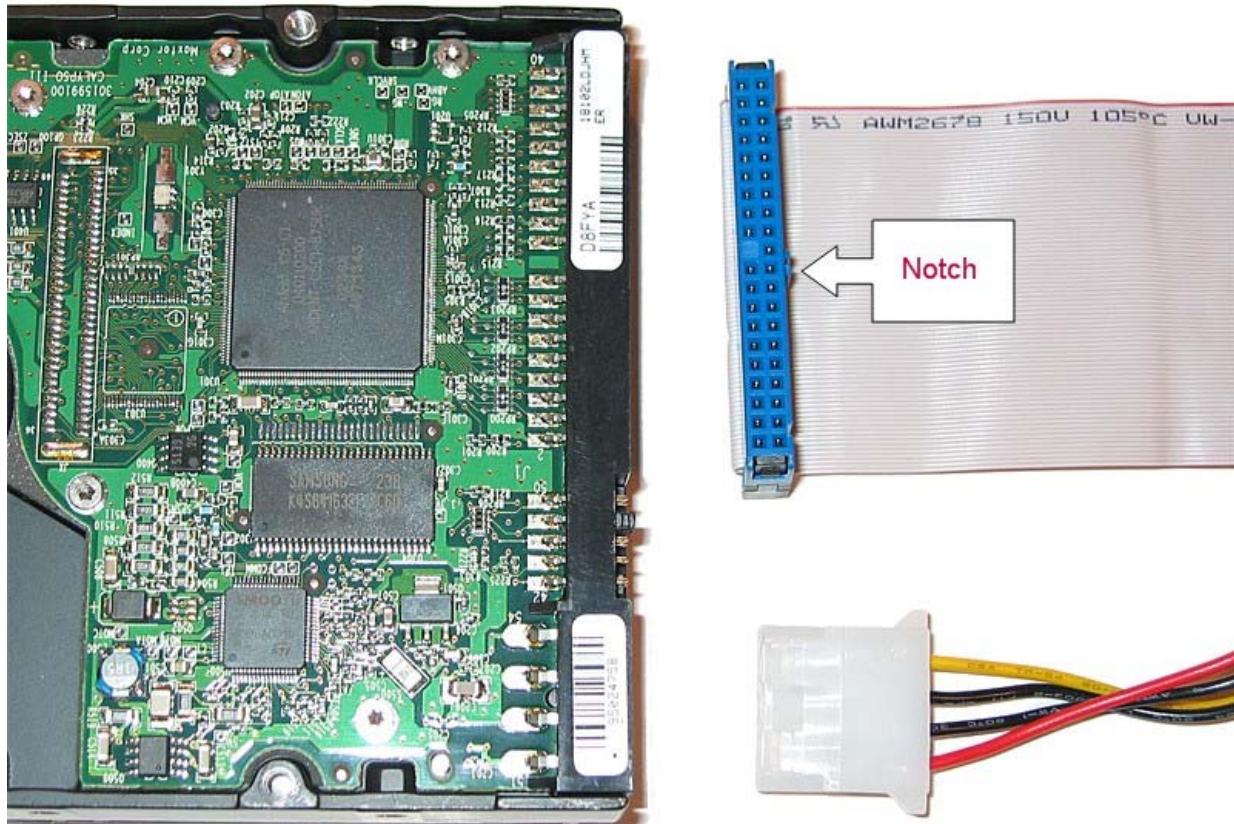
Over the years, disk interfaces have changed radically—in fact, there has been substantially more change in disk interfaces than in the disks themselves. Each has its benefits and drawbacks. There are currently three popular interfaces in widespread use: Integrated Drive Electronics (IDE), Small Computer System Interface (SCSI), and Serial Advanced Technology Attachment (SATA).

### *IDE*


IDE (also frequently called ATA) is an older and well-established disk interface. Most computers support IDE, and a variety of low-cost drives are available with this interface. IDE has developed over the past several years and now supports increased data throughput speeds.

The main failings with IDE drives are their limited throughput and cumbersome cabling. The throughput of IDE has been increased over the years with backward-compatible hardware upgrades. Today’s IDE drives can sustain 133MBps, and that speed is shared among all devices attached to the same IDE connector. Although this speed is faster than the original IDE devices, it is far slower than the now-common 3GBps available with SATA.

Cumbersome cabling has always plagued IDE. It is most often implemented as a flat ribbon cable as shown in Figure 1.3.



**Figure 1.3:** An IDE drive with data and power connectors.

 The notch must be oriented in the proper direction for the drive to work (it will fit on the wrong way in some configurations). Also of note is the bulk of the cable itself. This type of cable is not conducive to proper computer ventilation and can contribute to system overheating, causing hardware failures or erratic behavior.

## SCSI

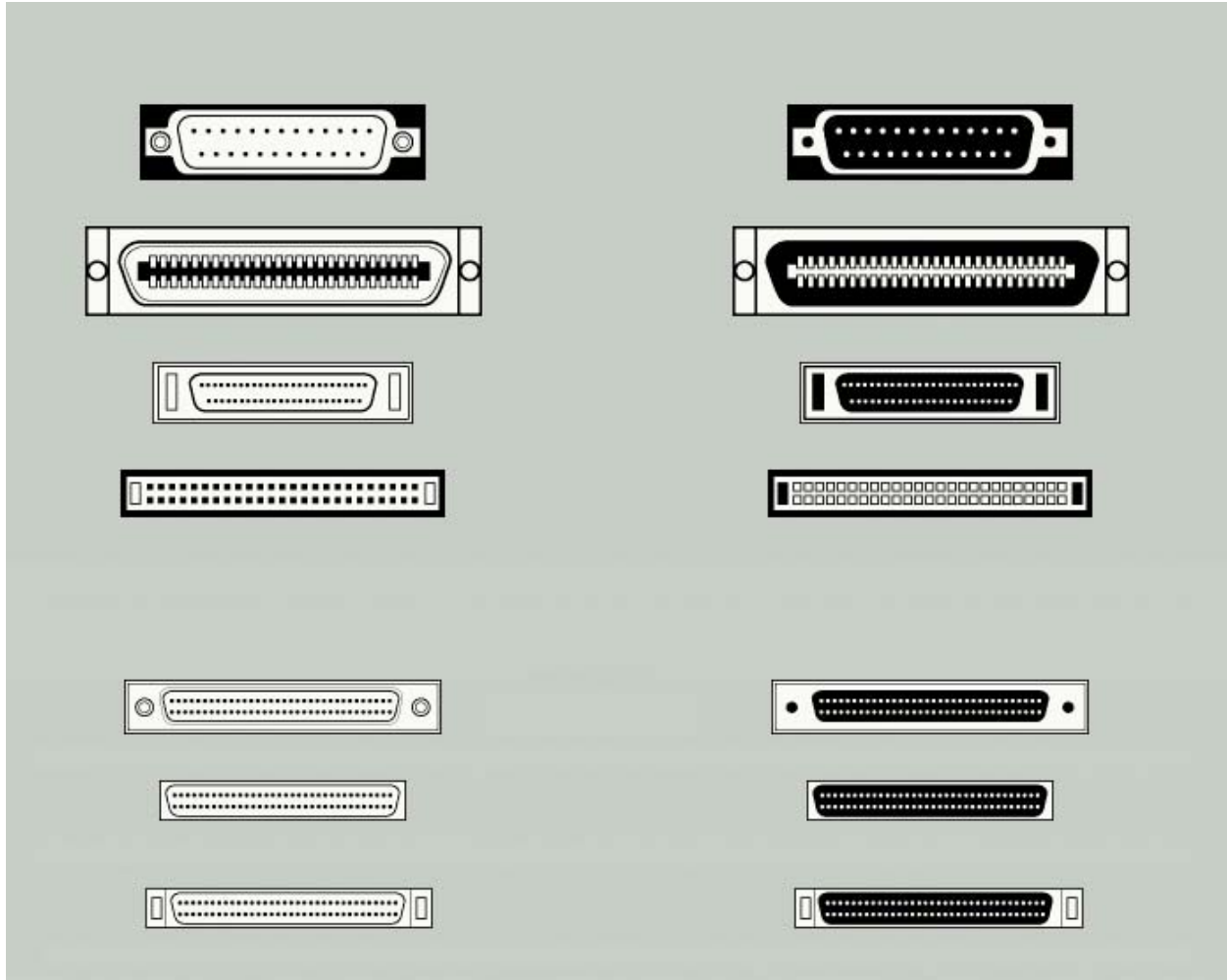
SCSI (pronounced skuzzy) disk drives have been available for decades. Its strengths include relatively fast throughput speeds, the ability to connect multiple disks to one disk interface, and automatic error recovery in most SCSI-enabled disks. SCSI has been popular for periods in the server, PC, and Macintosh segments of the computer industry.

Unfortunately, SCSI has many shortcomings. Primary among these is its constantly changing connectors and standards. Since the introduction of SCSI in 1986, it has undergone numerous revisions and updates. Nearly each update changes the connector configuration, requiring different cables and adapters to work properly. To get an idea of just how variable SCSI is, consider the following partial list of the major SCSI versions:

- SCSI-1
- SCSI-2
- SCSI-3
- Ultra-2
- Ultra-3
- Ultra-320
- Ultra-640
- iSCSI
- Serial SCSI

The complexity of the changing standards and different incompatible hardware revisions make management of SCSI devices difficult. For example, an existing investment of SCSI-3 cables and connectors is incompatible with new Serial SCSI investments.

It is also difficult for the IT professional to recognize the various SCSI connectors on sight, forcing most to carry references or refer to dedicated Web sites to identify hardware. For example, Figure 1.4 shows a diagram of a small subset of SCSI-1 and SCSI-2 connectors.



*Figure 1.4: A sample of SCSI-1 and SCSI-2 connectors.*

SCSI has also historically been a more expensive investment than other disk interfaces. Its complexity and requirement for advanced controller software often drive initial investment prices far beyond other similar technologies. Combine this with the fact that newer, simpler, and cheaper alternatives are available, and you'll understand why widespread use of SCSI-based devices is currently waning.

## SATA

A newer evolution of IDE is SATA. It has evolved as hardware engineers examine the strengths and failures of previous interfaces. In this way, it has an advantage over all other standards because it can improve on weaknesses while continuing to build on strengths.

SATA has a much simpler and smaller connector than either IDE or SCSI. Although SATA connectors are often somewhat fragile plastic connectors, they are engineered to meet the needs of a normal volume of connections and disconnections. Figure 1.5 shows a typical SATA connector.



*Figure 1.5: A SATA connector.*

SATA was also designed to be cost effective. Both the interface electronics and cabling can be produced very inexpensively and can easily coexist with an IDE interface on the same computer. This reality has helped drive widespread adoption of the standard.

Another benefit of SATA is its greatly enhanced throughput and optimized data transmission. Typical SATA speeds begin at 1.5GBps, and newer standards are already in place (with similar hardware) that provide 3GBps throughput. Currently, most new high-end computers are equipped with SATA drives.

### **Disk Interface Wrap Up**

The selection of a disk interface should be made on a cost-benefit basis. If the benefits of the more expensive formats outweigh the costs, that interface is the right one. You should also take into consideration current and future ownership costs, such as the cost of later disk expansion and the organization's current and future storage needs. However, none of these interfaces can be viewed as an absolutely "wrong" selection.

## Fault-Tolerant Disk Systems

Computers are generally made up of numerous electrical circuits that show little or no wear over time. They don't wear out primarily because they don't move—the electricity moves, but the components do not show signs of wear from the electrical signals. However, unlike most other parts of a computer, hard disks contain numerous moving parts. These moving parts include disk platters spinning at thousands of revolutions per minutes and read/write heads traveling back and forth over the disk. These high-precision components are designed within very tight tolerances and generally can last for years. But they do wear out and fail more often than other computer components simply due to their design.

Designers identified this potential weakness very early in the evolution of disk storage technology. They devised a standard rating, the mean time before failure (MTBF), to describe how long a disk with constant power and normal usage should last before it fails. This rating is somewhat arbitrary because of its prediction-based methodology, but it does help systems administrators discriminate between drive features. It also reminds administrators that disks are prone to failure and that measures should be taken to mitigate this risk.

A very popular disk-failure risk mitigation technique is to configure more than one disk to store the same data. For example, if a customer database is stored on a single hard disk, when that disk fails, the data is inaccessible. However, if that same data is stored on three hard disks, it is highly unlikely that all three disks will fail at the same moment. One failure does not render the data inaccessible. This disk redundancy method became very popular beginning in the late 1980s and continues its popularity today.

In 1988, a specific scheme that uses multiple disks for data redundancy was defined. This scheme was called Redundant Array of Inexpensive Disks (RAID). RAID defines several levels of redundancy using a number of disk drives. What sets RAID apart is that in general the redundancy is handled by hardware, such as a RAID-compliant drive chassis or a RAID-enabled disk controller. The redundancy provided by hardware-specific solutions can be very fast and enable online disk recovery operations that would not be possible if an OS or other software-based solution was used.

Some of the popular RAID levels include RAID 0, 1, and 5. Many RAID schemes employ either these levels or a combination of these and other redundancy schemes. Understanding these three key schemes will help you understand RAID overall.

### **RAID 0**

RAID 0 is also known as a striped set. This level of RAID writes data across exactly two physical disks with no redundancy or parity information. The loss of one disk results in the loss of all data, as there is no method for recovering data. For that reason, RAID 0 is actually not redundant at all and is not used where data redundancy is required. However, it is often mentioned as a RAID level. RAID 0 is frequently used in high-performance computer systems as a method to increase disk throughput speeds.

## **RAID 1**

RAID 1 is a true redundant scheme. Also known as disk mirroring, it is used to ensure the integrity and availability of data by making an exact copy of all data. In this scheme, exactly two disk drives are used. The drives are maintained as exact copies (or mirrors) of each other at all times. If one disk fails, the other can continue to function alone while repairs are made. Because the likelihood of both drives failing at the same moment is remote, this scheme is considered useful and many systems employ it.

However, the cost-per-byte of a RAID 1 implementation is relatively high compared with other redundancy schemes. For example, two 500Gb drives configured as RAID 1 yield 500Gb of accessible space—half is available, the other half is for redundancy. With the cost of disk storage continuing to fall (at the time of this writing), this is not usually a cause for concern.

## **RAID 5**

Although RAID 1 provides excellent data redundancy, it has a high cost per byte. RAID engineers looked for a way to reduce the cost of disk overhead while still providing redundancy (remember, this was done when disk drives were still very expensive). They came up with a scheme called RAID 5, also known as a striped set with parity.

In RAID 5, three or more disks are used. Data is written in blocks (stripes) across all but one of the disks. On the disk on which data is not written, parity information (also called a checksum) is stored. RAID 5 is often implemented in hardware in the form of smart disk controllers or smart RAID enclosures, so the computer and OS do not have to perform the complex parity calculations or disk management tasks.

When one disk in a RAID 5 array fails, the system continues as normal because the data from the lost disk can be calculated from the remaining data and parity information. System performance may temporarily decrease while that disk is down because of the extra operations performed, but that is more than made up for by the system uptime provided.

There are two big benefits of RAID 5. First, the failed disk can usually be replaced and initialized while the system is still online, virtually eliminating data downtime. Second, the cost per byte is much lower than that of RAID 1.

## **RAID Wrap Up**

There are a number of schemes available to guard against the fallibility of modern disk drives. RAID schemes are very popular and are often implemented in hardware solutions that partially or completely abstract the OS from the RAID details. These schemes can prove effective in increasing uptime, but care should be given as to which scheme is implemented to ensure that the appropriate level of data redundancy is achieved.

## File Systems

Disks store data using their own format, and the electrical connection between the disk and the computer has its own format. However, neither of these formats is conducive to easy use by application-level programmers or systems administrators because the formats are far too detailed and device-specific. Programmers and administrators need a way to logically organize, store, and retrieve data that is abstracted from the low-level mechanisms of data transmission and storage. File systems provide that layer of abstraction.

File systems are methods to store, organize, and retrieve data on a disk. They are often abstracted by the OS and made transparent to the user. For example, most Windows users cannot tell whether their file system is File Allocation Table (FAT), New Technology File System (NTFS), or High Performance File System (HPFS) unless they're looking for some specific feature only available on one of the systems.

There have been several significant file systems developed for the Windows platform. The most significant are FAT and NTFS. These file systems differ greatly in their capabilities and internal structure.

### **FAT**

When MS-DOS was first being developed, Bill Gates needed a basic file system to store and retrieve data. His development efforts led to the first version of the file system he called FAT in 1977.

FAT is an uncomplicated file system and was very appropriate for the era in which it was created. It stores data in a very basic format because computers of those days didn't need a complex hierarchical or extensible file system. It takes up very little space for itself because disk space was at a premium. Many features simply weren't considered because they weren't part of the thought process: robustness, error recovery, extended file descriptors, and security being good examples. None of these features were intended to be in Windows, so the file system had no need to support them. The file system was also not extensible, because at that time there was no concept of changing or extending the data that the file systems supported.

Many current administrators feel that FAT is a useless technology and should never be used. Although it is true that FAT isn't as advanced as other modern file systems, it certainly has its place in today's environments. For example, FAT is almost always used on removable media such as floppy disks and thumb drives. You can also use FAT for backward compatibility with other OSs in dual-boot scenarios, such as when you need to use MS-DOS and Windows NT on the same single-disk system. FAT comes in three distinct variations: FAT12, FAT16, and FAT32. The difference is in the number of bits used in their data addressing: 12, 16, and 32, respectively.



## FAT12

The oldest version of FAT is FAT12, which stores a maximum of 4077 files and supports up to a 32MB disk. Although this version was replaced by FAT16 for hard drive use as PC hard drives began to become available, FAT12 is still in use as the preferred format for floppy disks. Floppy disks have such limited space, and FAT12 can address it all with very limited overhead, making it an appropriate file system.

## FAT16

FAT16 is nearly identical to FAT12 except for its use of 16 bits in its addressing scheme. But this minor architectural change allows FAT16 to address hard drives up to 2GB and store up to 65517 files. FAT16 was very popular with MS-DOS and versions of Windows up to Windows 98.

## FAT32

In 1996, Microsoft recognized that hard drives were growing past the 2GB address limit of FAT16. The company addressed this problem by doubling the number of address bits to 32, creating a new file system called FAT32. This was first released in a service pack for Windows 95 and then Windows 98. This change allows FAT32 to manage hard drives of up to 2TB and store more than 200 million files. FAT32 is still in widespread use because it can manage current disk needs.

You do not need to know the detailed specifications of FAT. What you should remember is that FAT is in somewhat common use today. In general, disks that do not need to run FAT for a specific reason should be upgraded to NTFS eventually to get the numerous benefits of that advanced file system. But there are still several legitimate uses for FAT, and there is nothing fundamentally wrong with using it.

## NTFS

When Microsoft was developing Windows NT, they recognized that FAT was not capable of much future growth. FAT had a number of design limitations and was not extensible. Thus, the software architects began to develop a new file system from scratch. The file system they designed was NTFS and it premiered in Windows NT 3.1.

NTFS was an enormous step forward. It had a number of integrated features, including:

- Ownership attributes
- Security descriptors
- Metadata support
- Atomic operation with transactional logging
- Support for volume sizes up to 16 exabytes
- Support for international languages
- Extensible attributes

Although all these features were enormously beneficial, one that bears further mention is extensible attributes. Essentially, this feature allows a software developer to customize NTFS in the future without having to redesign the entire file system. For example, when Microsoft integrated file system encryption in Windows 2000 (Win2K), the company simply extended the functionality of NTFS. Doing so avoids costly changes or upgrades for programs and prevents broken functionality.

Although FAT was designed as a list of files on a hard drive, NTFS was designed as an extensible database that could store, among other things, files and directories. Thus, NTFS can be extremely efficient despite storing enormous amounts of data. It also means that NTFS can organize free and used disk space rather easily. It will become clear how important this is later in this guide.

NTFS is the preferred file system on Windows-based computer systems today. It is the default file system for all new drives. You should consider using NTFS whenever possible.

### **Other PC File Systems**

There have been other older file systems that were included with Windows in the past. In addition, many file systems have been ported to Windows over the years, including some that were never intended for use on a PC. Two are worth briefly mentioning, for very different reasons. One, HPFS, used to be supported in Windows NT and OS/2, so you might encounter it on rare occasions. The other, ext3, is not supported by any Windows version but is popular enough that you should be aware of its existence.

### **HPFS**

HPFS was designed to work with OS/2. It had a number of advanced features and was the file system of choice for OS/2 users, and was the preferred file system for volumes between 200 and 400MB (as this was its optimal operating size). Full support for HPFS was included in Windows NT 3.1 and 3.5 to both support upgrades from OS/2 servers and to support POSIX-based applications that required access to an HPFS volume. However, lack of use of this feature prompted Microsoft to remove the ability to create HPFS volumes and then finally all support for the file system.

It is rare to encounter HPFS-enabled computer systems today. Unless there is a critical need for maintaining HPFS on a system (for example, a critical application requires it), consider converting the volume to NTFS or upgrading the OS to a more current version.

### **ext3**

ext3 is the default file system for many Linux distributions. It is not officially supported on any Windows system. However, it is somewhat popular in the industry due to its inherent recoverability characteristics.

## File System Wrap Up

There are a number of file systems available. Many are older, inefficient on large modern disk drives, and only suitable for limited situations such as backward compatibility. For most Windows-based computers, NTFS should be the file system of choice.

## How Disks Are Used

At a very basic level, disks are written to and read from. But that level of understanding doesn't help you make decisions about how to manage storage. You need to probe a little deeper.

Let's take a look at a very common example. Suppose that SERVER01 is a Windows Server 2003 (WS2K3)-based file and print server in your company. On average, about 100 users have daily interaction with this server for data storage and retrieval and print jobs. SERVER01 is a top-of-the-line Xeon-based server with 8GB of memory and a 2TB disk array. The disk storage is configured as one non-fault-tolerant storage volume, and to address disaster recovery, nightly backups are made and sent offsite.

During an average work day, 400 print jobs are sent to SERVER01. The network bandwidth supports the data transfer from the clients just fine. When the print job is received by SERVER01, it is temporarily written to the disk array before being sent to the printer. Once the printer acknowledges receipt of the print job, the temporary file is deleted. This is the way printing works in Windows.

Also during the day, several hundred Microsoft Office files, such as Word documents and Excel spreadsheets, are accessed and edited on the server. Some files are just a few kilobytes in size, and others are quite large, as they contain graphics or video clips. During the normal operation of Microsoft Office software (and indeed most business software today), the files are saved to the server periodically as temporary files. These temporary files are placeholders to help recover an in-process document in the case of disconnection or a computer crash. It is not uncommon for tens or even hundreds of temporary files to exist on a heavily edited document. Once the file is saved and closed, all the temporary files it created are deleted.

In this small example, you can see that thousands of files are created, deleted, and edited throughout the course of a normal day on SERVER01. On the surface, this doesn't present a problem, as there is plenty of space and hard disk bandwidth. But if you look deeper, you'll see that there is the potential for significant stability and performance impact with this type of operation. Some of the disk-based performance considerations include fragmentation and I/O bandwidth.

## Fragmentation

Data is normally written to and read from hard disks in a linear fashion—one long piece of data. This is done to optimize disk reading and writing and increase performance. When numerous files are written and deleted, gaps in the drive's free space will appear. These gaps will affect future files because those files must fit into the gaps. If the files don't exactly fit into one gap, it will have to be placed into two or more gaps. This is fragmentation.

Consider that a hard disk is a huge surface that just stores 1s and 0s. There is only one way to read those 1s and 0s. The disk read/write head must move directly over the data and read it. If all the data for a file is in one small area, the read/write head may need to move very little or not at all to read the whole file. But if the data is scattered all over the disk, the read/write head needs to move a great deal to gather the data for the file. This difference can be negligible on small isolated files or an infrequently used system, but on a file server with thousands of files, the lag can quickly add up to a noticeable performance decrease.



One way to think of this is that the Earth is covered in tiny pebbles, and each pebble has a black side and a white side. Each pebble represents a bit of storage on a hard disk. The read/write head is a helicopter. Whenever you need data, the helicopter flies to the part of the Earth that has the file, lands, and the pilot reads out the position of each pebble to the requestor. When you're writing a file, the pilot must fly to the proper position, land, and begin flipping pebbles to their proper position according to the data being written. So as you can guess, having all the pebbles necessary for an operation together in one spot would save a lot of flying and landing.

Two common misconceptions about disk fragmentation are that newly installed computers are fragmentation-free and that NTFS eliminates fragmentation. The first idea, that new computers are unfragmented, is simply untrue. New computers can have extensive disk fragmentation. Often this is caused by numerous writes and deletes during computer installation. On computers upgraded from a previous OS, the problem is exacerbated because the drive may have been fragmented even before the upgrade began.

Although NTFS does actively seek to avoid fragmentation, it can only do so on a best-effort basis. If there is a readily available contiguous extent to store a new file, NTFS prefers that over a fragmented extent. But such extents are not always available and NTFS will provide any available disk space for storage, including fragmented space. There is no built-in functionality for NTFS to defragment or ensure the contiguous writing of files.

## Disk Bandwidth

One way to think of a computer is as a central data processor that reads and writes data. In that case, there are three performance considerations:

- How fast can I read data?
- How fast can I process the data?
- How fast can I write the data?

Most of the biggest computer breakthroughs and sales campaigns over the past several years have revolved around the processing consideration. The battle between Intel, AMD, and Motorola primarily revolved around processing power, not data bandwidth, because usually the different processor manufacturers can all use the same disk interfaces. Thus, as the computers beef up in the processor area, the data pathway becomes even more important.

Reading and writing data to RAM is relatively fast, as those data pathways are short and have few boundaries. But data access on long-term storage, such as hard disks, is different. The data must be converted to a different transmission method via a disk bus, such as IDE, SCSI, or SATA. This conversion takes a significant amount of time and resources compared with data access from RAM or a cache. Therefore, the disk access often becomes the performance bottleneck of a system.

## Summary

There are several factors that go into disk management. Disk interfaces, file systems, OSs, and other factors all play a part in disk performance and reliability. Selecting the right combination of these factors can play a key part in the behavior of your system. But no matter which selections you make, it is likely that you'll need to understand long-term disk management and maintenance issues. Chapter 2 will discuss those issues in detail.

### Additional Resources

For a more complete examination of disk architecture in Windows, see *Magnetic Storage Handbook* (McGraw-Hill Professional) by C. Denis Mee and Eric D. Daniel.

For more information about file systems, see *Windows NT File System Internals* (O'Reilly) by Rajeev Nagar or *Microsoft Windows Internals, Fourth Edition* (Microsoft Press) by Mark Russinovich and David Solomon.

For more information about fragmentation, see *How File Fragmentation Occurs On Windows XP / Windows Server 2003* at <http://files.diskeeper.com/pdf/HowFileFragmentationOccursonWindowsXP.pdf>. Chapters 2, 3, and 4 will also provide information about fragmentation.

## Download Additional eBooks from Realtime Nexus!

Realtime Nexus—The Digital Library provides world-class expert resources that IT professionals depend on to learn about the newest technologies. If you found this eBook to be informative, we encourage you to download more of our industry-leading technology eBooks and video guides at Realtime Nexus. Please visit <http://nexus.realtimepublishers.com>.

---

## Glossary

### Bottleneck

A part of a process that causes the entire process to slow.

### CD-ROM File System (CDFS)

An older file system used to store data on a compact disc.

### Cluster

The basic disk allocation unit for a file system. It is made up of one or more physical disk sectors.

### Compression

In NTFS, attempting to make files take less room on a disk by compressing data in the file. This is also called file compression.

### Defragmentation

The act of reallocating data storage on a disk such that each file's data is located in as few contiguous data runs as possible.

### ext3

A file system normally associated with the Linux operating system.

### Extent

A physically contiguous group of disk clusters used by a file as a single storage unit. Also called a *run*.

### File Allocation Table (FAT)

An older file system most closely associated with MS-DOS and early versions of Windows. FAT came in several varieties depending on the disk addressing scheme: FAT12, FAT16, and FAT32. FAT is still in use on some older systems or where MS-DOS compatibility must be maintained.

### File system

A standardized scheme for storing and retrieving data from secondary storage. Common file systems include NTFS, CDFS, and FAT.

### Fragmentation

The state in which data is no longer stored contiguously. This can be file fragmentation, where the file on the secondary storage device is stored in discontinuous extents, or data fragmentation, where the data structure within the file is discontinuous or suboptimal.

### High Performance File System (HPFS)

An older file system primarily used on OS/2 and Windows NT systems where OS/2 compatibility is required.

**Integrated Drive Electronics (IDE)**

An interface specification for connecting storage devices (such as hard disks) to computers.

**Metadata**

Data structures that describe the data stored in an NTFS volume. Metadata in NTFS contains several different files and data structures, most notably the MFT.

**Master File Table (MFT)**

The special metadata file on NTFS volumes that describes where all the resources on that NTFS volume are located, such as which clusters contain which directories and files.

**New Technology File System (NTFS)**

A transactional file system that supports metadata storage. Most newer Windows systems use NTFS as their default file system.

**Pagefile**

A file on secondary storage dedicated to storing data temporarily transferred (or paged) from main memory. A pagefile acts as a sort of extended RAM.

**Run**

See *extent*.

**Serial Advanced Technology Attachment (SATA)**

A newer version of the IDE disk interface standard that offers faster throughput and simpler connection options than IDE.

**Small Computer System Interface (SCSI)**

A popular disk interface best recognized for its high throughput and flexible disk connection options.

**Secondary storage**

Long-term persistent data storage device, most commonly implemented as a hard disk. This differs from primary storage, which is usually a short-term fast but volatile storage method such as RAM.

**Sector**

The basic unit of hard disk storage. Normally sectors are not addressed individually but are grouped into clusters by the file system.

**System files**

A specific type of file used by the operating system to provide critical elements of system functionality. System files are often treated specially by the file system.

**Temp files**

Temporary files created by many applications during normal data processing applications. Temp files are usually created for a short period of time and then deleted.