realtimepublishers.com™

The Definitive Guide™ To

# Exchange Disaster Recovery & Availability

XOsoft™ JUST KEEP WORKING

Paul Robichaux

# Introduction to Realtimepublishers

**By Sean Daily, Series Editor**

The book you are about to enjoy represents an entirely new modality of publishing and a major first in the industry. The founding concept behind Realtimepublishers.com is the idea of providing readers with high-quality books about today's most critical technology topics—at no cost to the reader. Although this feat might sound somewhat impossible to achieve, it is made possible through the vision and generosity of a corporate sponsor who agrees to bear the book's production expenses and host the book on its Web site for the benefit of its Web site visitors.

It should be pointed out that the free nature of these publications does not in any way diminish their quality. Without reservation, I can tell you that the book that you're now reading is the equivalent of any similar printed book you might find at your local bookstore—with the notable exception that it won't cost you $30 to $80. The Realtimepublishers publishing model also provides other significant benefits. For example, the electronic nature of this book makes activities such as chapter updates and additions and the release of a new edition possible in a far shorter timeframe than is the case with conventional printed books. Because Realtimepublishers publishes our titles in "real-time"—that is, as chapters are written or revised by the author—you benefit from receiving the information immediately rather than having to wait months or years to receive a complete product.

Finally, I'd like to note that our books are by no means paid advertisements for the sponsor. Realtimepublishers is an independent publishing company and maintains, by written agreement with the sponsor, 100 percent editorial control over the content of our titles. It is my opinion that this system of content delivery not only is of immeasurable value to readers but also will hold a significant place in the future of publishing.

As the founder of Realtimepublishers, my *raison d'être* is to create "dream team" projects—that is, to locate and work only with the industry's leading authors and sponsors, and publish books that help readers do their everyday jobs. To that end, I encourage and welcome your feedback on this or any other book in the Realtimepublishers.com series. If you would like to submit a comment, question, or suggestion, please do so by sending an email to feedback@realtimepublishers.com, leaving feedback on our Web site at http://www.realtimepublishers.com, or calling us at 800-509-0532 ext. 110.

Thanks for reading, and enjoy!

Sean Daily
Founder & Series Editor
Realtimepublishers.com, Inc.

## Copyright Statement

# Chapter 1: Basics of Exchange Protection

Understanding Exchange disaster recovery can take a while. The first step, naturally enough, is to gain an understanding of the fundamentals of Exchange protection. Later chapters will describe the specific technologies you can use to improve the availability and recoverability of your messaging system and its related business services. First, we must explore the causes of downtime, the general classes of availability and protection technology, and ways to assess your Exchange environment to determine the most effective methods for you to better protect your Exchange infrastructure.

## Where Does Downtime Come From?

IBM used to run a terrific television commercial for their servers; it used a sports motif and the catchphrase "There's no showtime without uptime." That slogan reflects what most of us think of when we consider the concept of downtime. However, a more accurate definition of availability can help us focus on the areas within a messaging deployment in which we can make the fastest improvement for the least investment.

### *Planned vs. Unplanned*

The first, and greatest, principle of downtime is that *some downtime is planned*. Some kinds of maintenance can't reasonably be done while maintaining normal service levels. For example, when you install an Exchange service pack, you must stop and restart the Exchange services; in the same vein, Windows service packs usually require that you reboot the server after applying the service pack. As an additional complication, third-party tools such as monitoring software, SAN management tools, and backup software may require additional maintenance—or even uninstall and reinstall cycles—when you deploy new service packs. To overcome these problems, smart organizations build in planned downtime or maintenance windows during which they can do this sort of work without impacting their overall availability numbers. Some vendors are disingenuous when they post uptime claims for their services or products; whenever someone quotes an uptime of 99.9 percent or better, you should carefully examine the claim to see whether they include planned downtime or *only* unplanned downtime.

The distinction between planned and unplanned downtime is important for several reasons. The biggest reason is simple: you have control over when your planned downtime happens. If you're not taking advantage of planned maintenance windows, when *are* you doing all of the maintenance that has to be done? For example, consider patches and fixes. You can't go wrong by planning time for patching. Given Microsoft's history of regularly issuing security patches and hotfixes (and the company's efforts to be more consistent about delivery dates for patches and service packs), you can count on having patches to install at least monthly. Not all of these installations will require reboots, but many will. Another reason why planned downtime is your friend: even really good hardware sometimes needs maintenance, replacement, or other work that will take it out of service. Of course, the same can be said of infrastructure services; it's not uncommon in large metropolitan areas to have scheduled power outages two or three times a year.

Having said that, there are things you can do to mitigate the impact of planned downtime on your application availability. Many of the protective measures described in this book are very effective at letting you carry out maintenance operations without noticeable impact on your users.

Unplanned downtime, of course, is what every administrator fears. Consider the case where you want to install a second CPU in an existing server that is otherwise running just fine. You shut down the server, pop in the CPU, and restart the server. So far, that time you've used is planned downtime…until the server fails at boot. Now you've abruptly crossed the threshold into unplanned (and unwanted) downtime. A more common example occurs when you discover that a mailbox database has somehow become corrupted and you have to reload it from a backup. During that recovery time, if your users can't access their mail, you're experiencing unplanned downtime. Every systems administrator has war stories about their worst unplanned-downtime experience, but you probably don't need to hear all of them to know what kind of impact unplanned downtime can have on your environment.

## Application vs. Server Availability

Another distinction that bears exploring is the difference between application availability and server availability. It's fairly easy to compute the uptime of a given server on Windows Server 2003 (WS2K3) by using the Shutdown Event Tracker; on older versions, you can check for event ID 6009 (logged during every boot) and 6006 or 6008 (indicating clean and dirty shutdowns). However, this data only tells you how much time your physical server was up—server availability. If the server's up, but the *services* it offers aren't available, you can make a pretty good argument that the server is useless. For an Exchange server, if the information store or system attendant services are down, or if one or more mailbox databases are dismounted, users on those databases can't send or receive mail. You can have server availability without application availability, but the reverse is not true. Thus, throughout the rest of the book, "availability" refers to the availability of the *applications* that run on a server—particularly, that means Exchange (or the infrastructure services on which it depends).

More broadly, this distinction raises the question of what it means to say that your Exchange services are available. If you have five servers, each of which holds 2000 mailboxes, and one server fails, your messaging operations can credibly be described as 80 percent available. However, for the unfortunate folks on the failed server, messaging is 0 percent available. Generally, most organizations calculate downtime for individual servers or services, then use those figures to derive the aggregate availability for a class of application. This approach requires you to keep more detailed availability statistics, but it also gives you much better visibility into exactly which servers and services are failing most often—something you need to know in order to fix things.

### *Counting the Nines*

Before moving forward, let's talk about how to actually measure downtime. In their seminal book *Blueprints for High Availability* (John Wiley and Sons, 2003), Marcus and Stern presented a table that converts the percentage of uptime (colloquially known as the "number of nines") into actual amounts of downtime. The results, which Table 1.1 shows, are enlightening. Let's start with the 98 percent mark; in 2001, the Gartner Group claimed that the average uptime for a Windows server with no special hardware or software features was about 98 percent—which translates to about 3.3 hours of unplanned downtime per week on average. Clearly most of us have better uptime than that on our Exchange servers today, at least on a weekly basis!

| Percentage Uptime | Percentage Downtime | Downtime Per Year | Downtime Per Week |
|---|---|---|---|
| 98% | 2% | 7d 7h | 3h 22m |
| 99% | 1% | 3d 15h | 1h 41m |
| 99.8% | 0.2% | 17h 30m | 20m 10sec |
| 99.9% | 0.1% | 8h 45m | 10m 5sec |
| 99.99% | 0.01% | 52.5m | 60sec |
| 99.999% | 0.001% | 5.25m | 6sec |
| 99.9999% | 0.0001% | 31.5sec | 0.6sec |

***Table 1.1: Turning uptime numbers into calendar time.***

Of course, if you look at the "downtime per year" numbers, things change a bit. At 99.9 percent availability, you experience a little more than 8 hours of downtime per year. For many organizations, it takes longer than 8 hours to restore the contents of a server from tape, so hitting three nines is only possible if a tape restore isn't required over the course of an entire year. Very few organizations have hard business requirements for continuous, 365-day operations. In fact, at most businesses, there are probably lots of times during the year when planned or unplanned downtime would have absolutely no business impact. For example, ask yourself whether, in your current environment, it's really necessary to maintain 99.9 percent uptime on weekends and holidays? For most businesses, the answer is "no," and the cost savings that come about from that decision can be significant. If you can afford to tolerate downtime, you can probably make do with a less expensive, less complex set of disaster recovery and continuance plans.

Back to planned and unplanned downtime. Most organizations make a distinction between "pure" uptime and uptime *excluding planned downtime*. This distinction is important. If you say "My Exchange servers had 99.99 percent uptime last year," that raises the reasonable question of when I did necessary hardware or software maintenance on them. The first time you must install a large update (for example, WS2K3 SP1), you'll probably blow through at least an hour per server. With a dozen servers, in the aggregate, you'll end up with as much as 12 hours of patch-related downtime, although given enough IT staff, you can reduce that by upgrading more than one server at a time. If this time is charged against your total availability budget, it is unlikely that you'll hit 99.9 percent (although 99.8 percent might still be within reach). However, if that time is charged against your budget for planned downtime (which hopefully takes into account times when your business operations don't require messaging capability), you can still credibly claim 99.9 percent, or better, of *planned* uptime.

### Common Causes of Planned Downtime

It might seem a little odd to talk about causes of planned downtime, but it's actually a useful exercise. Many companies needlessly take unplanned downtime to do things that should really have been included in their planned maintenance procedures. Common operations that should be undertaken during your planned downtime windows include:

- Hardware maintenance, including planned repairs, upgrades, and physical moves of servers from one location to another—Taking time to do scheduled maintenance on your hardware will help reduce the chances that you'll have to do *unscheduled* maintenance on it during an outage. Of course, modern server-class hardware usually includes redundant power supplies, fans, and disks, and many models include additional redundancy for network cards and even RAM. However, depending on your hardware, if a non-critical component fails, you might be able to wait to swap it out until a planned downtime occurs.
- Software maintenance, including installing operating system (OS) or application patches or fixes—Microsoft has made a great effort to deliver fixes that don't require reboots or service restarts, but in some cases reboots are still required. Don't forget to include time for updating or patching third-party software, including antivirus scanners, backup tools, and other programs that receive periodic updates from their vendors.
- Database maintenance, including using *eseutil* and *isinteg* to perform integrity checks or to defragment a mailbox or public folder database—The Exchange information store already performs several necessary maintenance tasks as part of its regular online maintenance process; although you can schedule when they run, you can't (and shouldn't) prevent them from doing so. In addition to these tasks, you may sometimes find it necessary to defragment your databases or to copy them to an alternative server and run integrity checks on them.

- Offline backups—There are many backup utilities on the market that support the Exchange Extensible Storage Engine (ESE) backup interfaces; however, sometimes there's no substitute for an offline backup, in which you make a complete copy of the databases and log files while the information store has them dismounted. Of course, dismounting the databases (or stopping the IS) renders those databases' data inaccessible, which is why it's normally done only during a planned maintenance window. (An exception is when you want to replicate the databases or log files, in which case, some solutions work best if you dismount the databases before creating the replicas.)
- Infrastructure maintenance—It's not uncommon to have planned outages so that a building owner can work on the electrical or HVAC systems in the building. Although you may not have control over the timing of these outages, most building management companies and public utilities publish a schedule of outages well enough in advance for you to make appropriate plans.

## *Common Causes of Unplanned Downtime*

Unplanned downtime is, by definition, unwanted downtime. Understanding the most common causes is the first step toward being ready to do something about them. A few of the items highlighted in the following list can not only cause downtime but also make an outage last longer than it should have.

### Human Error

Perhaps unsurprisingly, *administrators* are significant causes of unplanned downtime. Estimates of human error–caused downtime vary widely, ranging from 15 percent (Sun Microsystems) to 40 percent (the Gartner Group) to 59 percent (a *NetworkWorld Fusion* study of downtime in routed IP networks). There are lots of ways for administrators to accidentally cause downtime in messaging systems:

- Physical issues such as spilling coffee into a storage rack, unplugging a critical piece of network equipment, or improperly installing Fibre Channel cables in a SAN—Bonus points are awarded for physical actions that permanently damage things, such as forcing a keyed connector in the wrong way or trying to hot-swap disks on a storage device that doesn't support hot swaps.
- Accidental configuration changes, either to Exchange itself or to infrastructure services—An example would be typing the wrong address for a Simple Mail Transfer Protocol (SMTP) smart host into the SMTP virtual server properties dialog box, but there are hundreds of others, many of which revolve around DNS and Active Directory (AD).
- Accidental deletion of important data—In Exchange, this most often happens when administrators delete a mailbox that someone actually needed; downtime-related instances of this problem usually involve deleting logs or databases, or system files that Windows or Exchange requires.

- Improper procedures, the most common example of which is probably failure to check the event log (and media) to ensure that backups are working properly. This may not sound like a downtime issue, but it can quickly become one when it's necessary to restore and no usable backups exist!
- Poor decision-making brought on by insufficient knowledge or planning, like deciding to install a Windows service pack in the middle of a workday or deleting transaction log files manually to free up disk space—A great example: administrators who install non-Exchange-aware antivirus tools, then scan the Exchange databases and transaction logs with them, causing corruption of the store and databases.

These kinds of errors are relatively easy and inexpensive to protect against; the first step, naturally enough, is to make sure your administrators are well-trained so that they don't commit these types of mistakes.

### Disasters

Disasters are greatly feared; after all, that is how we got the common term "disaster recovery." By *disaster*, I mean large-scale events such as hurricanes, tornadoes, or other occurrences that can't reasonably be predicted and are difficult to protect against beforehand. Depending on where your business operations are located, your organization may be at risk from weather, fire, flooding, excessive snowfall, earthquakes, tsunamis, or some combination. Predicting these events, and planning to handle them, can be very difficult; fortunately, most of the same measures that offer protection against lesser problems are effective at mitigating disaster risk.

### Hardware Failure

When you think about it empirically, modern server-class computer hardware is both astonishingly inexpensive and incredibly reliable. A $1500 server from a large hardware company such as Dell or Hewlett-Packard is likely to include redundant power supplies, redundant fans, a redundant storage system, and an array of monitoring software that would have thrilled purchasers of a $500,000 IBM mainframe just a few years ago. However, that doesn't mean that the hardware never fails—remember, the stuff you buy is by and large made of components that are supplied by the lowest bidder (as evidenced by the rash of capacitor failures that struck both Apple and Dell in early 2005). In addition, your servers may not all have the same level of protection, particularly if you've built your own servers or bought them from a local white-box provider.

There are several critical hardware items in your servers that don't usually have redundant protection, including:

- The CPU—The failure rate for Intel and AMD CPUs is exceptionally low, but when a CPU fails, the odds are good that your server will stop working (of course, that depends on how many CPUs you have and on the exact nature of the failure).
- The motherboard; a failed component, faulty RAM socket, or burnt trace can knock your server out of commission.
- RAM—Error-correcting (ECC) RAM is pretty common in medium-range servers, but a faulty DIMM can still cause problems ranging from inconsistent operation to complete failure.
- Peripherals, whether they're on PCI cards or attached externally—Some servers have hot-swap-capable PCI slots, but most don't, so failures of video cards, FC host-bus adapters (HBAs), and the like may result in an unplanned outage for replacement.

### Disk and Storage Failure

If you asked administrators what kind of failures they were most concerned about, my guess is that a large majority would cite the storage subsystem on their servers. As with server hardware, modern hard disks are extremely reliable. However, the nature of their design means that they have moving and spinning parts, and the incredibly small operating tolerances required for them to work means that there is little margin for error. The biggest source of storage-related failures are the disks themselves; any of the electromechanical components (such as the motor, the actuators that position the read/write heads, or the heads themselves) can fail in a cacophony of screeching aluminum, or the onboard electronics that interface the disk to your storage bus can silently fail. Either way, when a single disk dies, the impact varies according to the kind of storage subsystem you're using and the data that's stored on it.

Later chapters will examine various storage designs to see how they hold up against Microsoft's recommendations.

**The Meaning of Mean Time Between Failures**

You've probably seen hardware manufacturers talk about the Mean Time Between Failures (MTBF) for their equipment. For example, disk drive manufacturers often claim MTBFs of several hundred thousand hours. This statistic sounds great (and it is, compared with hardware MTBFs from 10 or 15 years ago), but if you have more than a handful of servers or disks, you'll quickly find that the MTBFs have to be carefully considered. In fact, it helps to know how disk manufacturers calculate the MTBF in the first place—they take a batch of drives (several hundred to a few thousand) and test them under fixed environmental conditions. When the first drive in the batch fails, they use the test run time to calculate the MTBF. Let's say that there are 1250 disks in the batch, and they run for 28 days before the first one fails. The MTBF can thus be cited as 28 * 24 * 1250 = 840,000 hours. Remember, this is the *average* time between failures; it's not a guarantee.

What does this mean for your availability? Say that you have ten servers. Each server has a dozen disks, the MTBF of which is 100,000 hours each. Simply put, that means that you can expect any *individual* disk to last for 100,000 hours, or about 11 years. However, the MTBF of any one of your servers is actually the *sum* of the MTBFs of its components: twelve disks, each with a 100,000-hour MTBF, means that on average you can expect one drive to fail every 100,000/12 = 8333 hours, or slightly less than once per year. As you increase the number of servers and disks (and as you factor in other electromechanical components, such as fans and power supplies), you can see that adding servers and disks actually increases the odds of a failure.

Of course, the storage system is more than just a bunch of disks (JBOD), at least in most cases. Other storage components may fail; depending on the storage system that you're using, the impact of these failures can range from minor to catastrophic. Examples abound:

- Many administrators depend on battery-backed Redundant Array of Independent Disks (RAID) cards for extra data protection. If the battery—or, worse still, the controller—fails, this failure usually spells doom for data on the disks. Depending on the controller type, you may or may not be able to actually plug the disks into another controller and reuse them.
- In Fibre Channel (FC) storage area networks (SANs), losing a FC switch can cause problems ranging from temporary spikes in I/O latency to outright data loss, and there isn't a good way to predict in advance which of these results you'll get. The same thing is true to a slightly lesser degree of iSCSI SANs; there's no fibre, but network interruptions can cause the same effects.
- No matter what kind of storage interconnect system you use, damaged, cheap, or improperly attached cables can cause electrical or optical noise on the data bus, which can in turn lead to data corruption or outright failures.

## Software Failure

Software failures aren't usually lumped in the same category as the other causes of unplanned downtime, but they should be; software faults can certainly cause service interruption, data corruption, and all the associated downtime costs. I don't remember the last time I saw a data corruption failure that could be directly attributed to Windows or Exchange; it's more common to find these problems caused by faulty device drivers or firmware, virus scanners, or other third-party products that access Exchange databases directly. However, even these problems are fairly rare. The most common class of software failure is probably made up of those errors related to Windows system files; for some inexplicable reason, files can occasionally be removed or damaged even while Windows File Protection (WFP) is active, to predictable effect.

## Network and Infrastructure Failure

The big catch-all class of failures is network and infrastructure failure. As I was writing this chapter, I got an alert notifying me of unusual queue growth on my Barracuda spam firewall; it turns out that the T-1 line to my office in Redmond had failed, so mail was stacking up on the Barracuda box. This classic example is an illustration of an infrastructure failure, but there are lots of other examples—on average, there seem to be a few major network-related outages each week in various parts of North America. Some are the result of physical damage to cabling; others are caused by equipment failure at Internet service providers (ISPs), and some are caused by routing or configuration problems (or mistakes) along the route between points A and B.

The common thread these failures share is that some infrastructure service, usually related to Internet connectivity, fails in a way that prevents people from accessing their messaging services. Partial failures may affect only some aspects of messaging service, such as inbound message traffic; more serious failures may cut off all external access to the servers that host your messaging environment. Whether these failures are serious depends on your business: if you depend on being able to exchange messages with customers, losing your Internet connectivity can have a serious impact that doesn't apply to sites where intra-organization messaging is more critical.

One interesting side note about network failures: as outsourced spam filtering providers become more popular, the quality and availability of the link between you and your filtering provider becomes increasingly important. This is even truer if you get high mail volumes that warrant a dedicated physical or virtual circuit to your hosted service; if that circuit dies, you'll need to make quick arrangements for mail delivery through an alternative route.

## The Availability Curve

Figure 1.1 shows a plot of availability and protection technologies. The horizontal axis represents cost, while the vertical axis represents availability, expressed as a percentage. This graph isn't scientific, so you could probably quibble about exactly where specific products from various vendors might be located, which isn't important. What *is* important is that this graph shows how these technologies can be stratified into four layers.

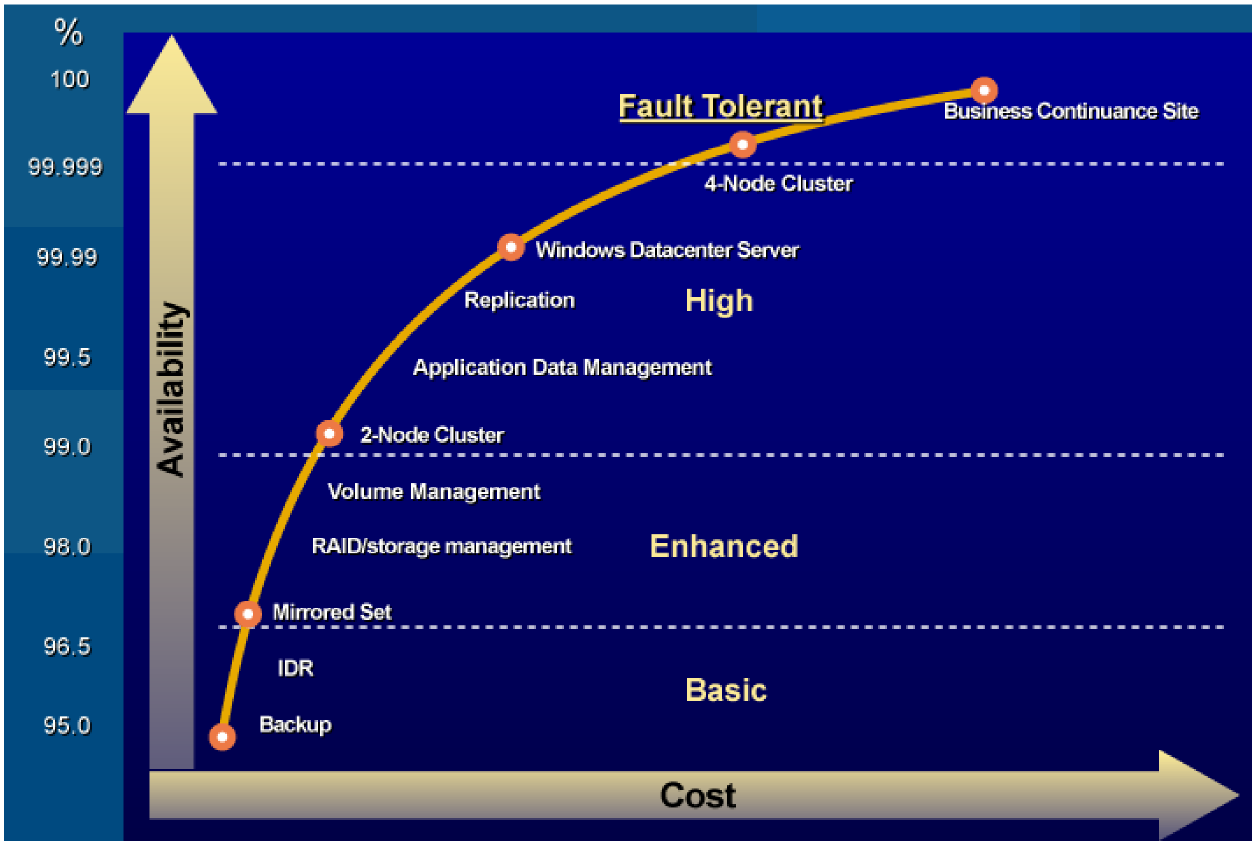

***Figure 1.1: The availability curve.***

Each layer has its own definition; throughout the rest of the book, I'll be grouping technologies and products using these definitions to make it easier to compare similar solutions:

- Basic availability is just that: basic. In this category, I include basic backup and recovery tools and so-called "intelligent" or "automatic" system recovery tools. Note that backup is the lowest point on the curve. The reason is that, strictly speaking, having backups doesn't do much for your availability. Instead, good backup and restore procedures cut your downtime *after* you have a failure; many of the other technologies on the curve provide proactive protection to keep the outage from happening in the first place.
- Enhanced covers the territory between basic availability and the 99 percent uptime level. Example technologies that fall into this category include mirroring, striping with parity, and volume management software that allows multiple-mirror sets.

- High availability begins at 99 percent and goes up from there. This layer is the one most administrators think of when they think seriously about raising their uptime; in this category, I'd include Microsoft's clustering technologies along with software- or hardware-based data replication and the selective use of the datacenter edition of WS2K3.
- Fault-tolerant systems offer more than just high availability; they can survive faults that would cause service or application outages on lesser systems. The computers that run the United States' telephone system are a great example of fault-tolerant systems, as are the computers on the space shuttle. These systems, which tend to be extremely expensive, have redundancy for virtually every component, all of which are hot-swappable. They also use a wide array of predictive monitoring and alerting techniques to alert administrators *before* a component fails, and their exotic hardware is combined with rigorous change- and configuration-management processes that help ensure the stability and good behavior of their software.

The exact placement of individual technologies on this curve isn't that important; within each level, individual vendors' capabilities and pricing make it difficult to draw a precise curve. What *is* important is the fact that there are multiple levels of availability, and that in each of those levels, you have some choices to make about which technologies to use and how much money to spend.

## What Does Protection Cost?

The idea that you can precisely quantify how much it costs you to protect against unwanted downtime is a good one, but (like many other good ideas) the devil is in the details. On one hand, it's easy to quantify what a particular protective technology costs: you merely have to ask the vendor to find out, although there are sometimes hidden costs that arise from infrastructure or operational requirements associated with a particular solution. It's also fairly easy to figure out what your budget allocates for disaster recovery, availability, and business continuance.

On the other hand, it can be quite difficult to figure out how much you *should* spend for availability. That decision has to factor in several variables, including what downtime actually costs your business and how much downtime you can afford to tolerate. For example, I once had an interesting discussion at TechEd with a gentleman who wanted to increase the availability of the messaging system at his company, which operated cement plants on both United States' coasts. The company's desired recovery time objective (RTO) was 7 days: in other words, if they had a failure of the messaging system in one plant, they would be happy if it could be restored within 7 days! An RTO of 7 days is an eternity by most standards of disaster recovery, but it was driven by the fact that their business could operate normally without their messaging system—something that isn't true of most businesses. As you consider what a reasonable RTO for your organization looks like, be sure to factor in management expectations, both positive and negative, that may affect what "reasonable" means in this context.

### Quantifying Risk

Actuaries and IT risk mitigation experts use a simple formula to calculate the degree of risk associated with different kinds of events (ranging from hurricanes to security penetrations):

*Risk = (probability of event) * (expected damage if event occurs)*

Thus, the risk of a given event is the product of the likelihood that the event will occur, multiplied by the damage that can reasonably be expected if it does. The formula thus takes into account both low-probability high-damage events such as earthquakes and higher-probability lower-damage events such as loss of a backup tape, extended power outages, malfeasance of a trusted administrator, and so on. The first step toward deciding what you can afford to spend on protection is to understand the risks your organization faces (at least the ones that pertain to messaging) and decide which ones are most significant.

Of course, any quantitative exercise like this raises the interesting problem of ties. Let's say that you have two events: a 10 percent chance of a $10 million loss or a 50 percent chance of a $2 million loss. The two events both have the same risk score, so how do you decide which one to protect against first? The cost of protection certainly is a factor; if you could protect against the $10 million loss for $100, you'd be foolish not to do so. In my experience, most businesses wisely choose to protect against the more likely risks they face, choosing what to protect first based on a combination of protection cost and the actual risk that is mitigated by the protection.

### Cost, Benefits, and Drawbacks of Various Technologies

Disaster recovery and high availability technologies have been with us since the early 1960s. Since that time, the capability and affordability of these technologies has gotten a great deal better, but it's still necessary to understand the general costs and benefits of each of these technologies.

## Backup

Backup systems are generally used as a last-ditch means of restoring data after a failure. In this role, they're a necessary part of any serious effort to improve messaging availability, even though the mere presence of a functioning backup system does nothing to raise your uptime—it's during an outage that the ability to quickly restore a reasonably up-to-date copy of your data becomes priceless.

Most backup systems use magnetic tape, which has the advantage of being generally cheaper than disks. Of course, you get what you pay for; tapes are less resilient, slower, and have lower storage densities than disks. To some extent, you can work around some of these disadvantages with appropriate hardware; a striped array of fast LTO tape drives, backed by a robot tape library, can store a jaw-dropping amount of data and retrieve it relatively quickly. Tapes provide a good archival mechanism because it's easy to take tapes to an alternative site, and there is a well-developed infrastructure of data protection companies who will sell you secure offsite space in whatever amounts you require.

Of course, there are other backup technologies besides tape; Figure 1.2 shows a pyramid that represents the access speed, media cost, and storage capacity of common backup technologies.

I haven't made any attempt to quantify the precise cost of any of these technologies; instead, the *relative* cost is more interesting for our purposes here.

Because the purchase cost of fixed disks keeps dropping, an increasing number of sites are choosing to do disk-to-disk backups (because they're fast), then move the backup files to tape for archival or offsite storage. Microsoft is pushing this trend with its Data Protection Manager (DPM) product, as are various backup vendors.

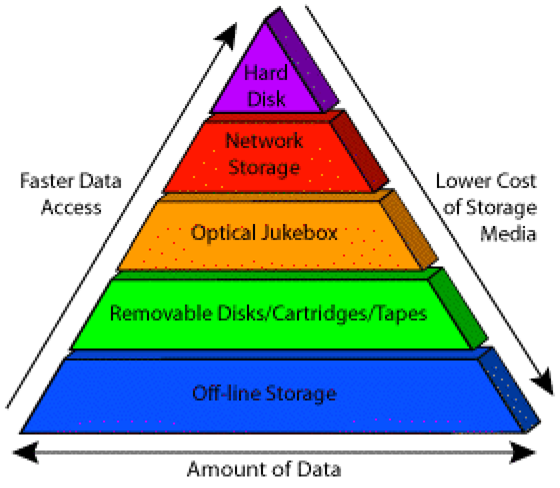

***Figure 1.2: The cost-vs-speed-vs-capacity pyramid for offline storage.***

## RAID

When David Patterson, Randy Katz, Garth Gibson, and Peter Chen developed the idea of RAID in the late 1980s, I suspect they had some inkling of what the future of storage technologies looked like—although it's fair to say that they might not have predicted how fast storage cost would drop. With the imminent arrival of perpendicular recording technologies, the day isn't far off when individual computers—not servers—will have multiple terabytes of local storage, and you can imagine what that kind of local storage capacity will do to the average storage allocation on servers. Just having more storage isn't sufficient, though, unless you don't care about what happens to the data. While backup technologies attempt to restore data back to complete and correct condition after a failure, RAID attempts to prevent or mitigate failures so that restore operations aren't necessary. Briefly put, RAID uses software or hardware to spread data across multiple disks; depending on how the data is spread, and what ancillary technologies are used, administrators can choose a RAID level that maximizes performance, redundancy, or a combination of both.

For a more detailed discussion of RAID levels and which ones are most applicable to Exchange, see Chapter 3.

The cost and performance of a RAID system depends on several factors; probably the biggest is the RAID level. Some RAID levels provide only redundancy. For example, with a mirrored (RAID-1) configuration, two or more disks maintain identical copies of the data, adding redundancy at the expense of additional disks (and adding a performance penalty for writes to the mirrored copies). Other configurations, such as the familiar RAID-5 (striping with parity) configuration, try to strike a balance between cost and redundancy, while still others (notably the increasingly popular RAID-1+0 configurations) add cost to provide faster speeds and better redundancy.

An example will help illustrate the cost tradeoffs. Say you have 100GB of mail data on a given server. You can store it:

- On a single 120GB disk, which will be inexpensive but will provide no redundancy nor performance benefits, if the disk fails, your data will be gone.
- On a mirrored pair of 120GB disks; this option costs twice as much as the single-disk solution but protects you against a failure of either of the two disks.
- On a RAID-5 set with at least three disks; in this configuration, you would spend three times as much to survive the failure of any one disk, but you would also get improved read and write performance and a total of 240GB of usable storage (120 * 3, minus the space allocated for parity).
- On a RAID-1+0 set, which would require a minimum of four disks (two mirrored pairs across which data can be striped)—out of the 480GB worth of disk, you only get 240GB of usable capacity, but you get much better protection against disk failure than RAID-5, with better performance to boot.

Depending on what kind of RAID software or controller you use, another potential advantage of RAID storage is the ability to reconfigure it on the fly. Being able to expand an existing array by adding drives is useful—all the more so if you can do it without having to take the array volumes offline. Most midrange servers now include onboard RAID controllers because the incremental cost is low and the performance and redundancy benefits are high.

### SANs

The basic idea behind SANs is simple: take many physical disks and couple them with a powerful controller that aggregates them into logical volumes, then make those logical volumes available across a dedicated high-speed connection. However, as with many other things that can be simply described, the reality turns out to be somewhat more complicated. SANs add significant cost and complexity—so why are they so popular? The answer is simple as well: they offer some very desirable benefits *if* they're properly designed, sized, deployed, and maintained. These benefits include:

- Better performance—A modern disk drive might offer the ability to perform 100 I/O operations per second (IOPS). An average Exchange user generates about 0.4 IOPS in normal operation, so a single disk can service about 250 users. If you want to put more users than that on a server, or if your users generate more traffic (say, 0.75 IOPS or more), you need more disks. SANs make it easy to offer extremely large IOPS rates: 50 disks, each of which can offer 100 IOPS, gives you 5000 IOPS. Of course, some of these I/O requests are reads, and some are writes; for writes, it's important to factor in the write overhead for RAID-1+0 and RAID-5.

Microsoft publishes a guide called Optimizing Storage for Exchange 2003 that details the process of designing a storage subsystem that provides the best possible performance and fault tolerance. See http://www.microsoft.com/technet/prodtechnol/exchange/guides/StoragePerformance.

- Better provisioning—One key driver behind SANs is that you can put all of your application data on the SAN, adding capacity as necessary. If your Exchange database needs more room, no problem—the SAN management tool makes it easy to add capacity by reallocating physical disks between logical volumes. As you consolidate data onto the SAN, what you're really doing is reducing the percentage of disk space that you've paid for but aren't using in your organization.
- Better resiliency—SANs offer extremely good fault tolerance; even "starter" SANs offer redundant connections from the host to the SAN, and many include redundant controllers and advanced management and monitoring software that can automatically trigger recovery from many kinds of failure.
- Good cluster support—SANs are particularly useful in clustered Exchange environments. Logical volumes on the SAN can be owned by only one cluster node at a time, but during failover, switching volumes from one host to another is simple.
- Advanced disaster recovery support—SAN-based solutions often support point-in-time copies and replication at the controller level, meaning that Windows and Exchange never need to be aware of what's happening to the underlying data. Microsoft's Volume Shadow Copy Service (VSS) provides a supported, safe way to take nearly instantaneous point-in-time copies of Exchange databases without corrupting or damaging the copy.

As I mentioned earlier, though, these benefits come at a cost. SANs are more expensive to buy and manage than direct-attached storage (DAS), and SANs require a great deal of specialized expertise to set up and maintain. Traditional SANs use the FC communications standard over either optical fiber or copper cables; FC SANs require HBAs for each computer that you want to have access to the SAN, as well as FC switches to link the devices on the SAN fabric. The recent introduction of the iSCSI SAN standard has created a booming market in low- to medium-cost SANs (at least relative to the cost of FC SANs) that offer many of the same benefits of SANs using TCP/IP over Gigabit Ethernet. Although iSCSI eliminates some of the cost and complexity issues associated with FC SANs, they still require some special care.

### Clustering

If you ask a random sample of Exchange administrators to name the first availability technology that pops into their minds, the most popular answer will be "clustering." Like SANs, clustering can add a significant degree of complexity and cost to Exchange deployments, but it also offers some benefits that may make it appropriate for your organization.

Microsoft's clustering support in Windows allows for 2- to 8-node clusters. Exchange Server 2003 supports a clustering mode known as N+1, where the N active servers are joined by one or more passive servers that can accept work from an active node that becomes unavailable, either due to a failure or due to planned maintenance. For example, Microsoft's Exchange deployment uses 7-node clusters: four active mailbox servers, two passive nodes, and one "utility" node that can be used for monitoring, management, and backup or restore operations.

The goal of clustering is to provide enhanced availability by moving services between nodes as necessary to maintain normal operations even in the event of a node failure. In this light, it offers some useful capabilities:

- Planned maintenance becomes trivial to implement—To update a node in a cluster, just fail its workload over to any other node in the cluster, then do your maintenance work on the node. Clustering makes it easy to perform hardware upgrades, apply service packs, and make configuration changes without disrupting users' ability to work.
- Unexpected failures become more survivable—Any hardware or software condition that would cause an individual server to fail will trigger a failover; assuming that the cluster's working properly, that failover will cause the failed node's workload to be assumed by another node, again without serious interruption of users' work.
- Clustering provides good scalability when combined with SANs—The combination allows easy reallocation of storage and mailbox resources as demand dictates.
- It's both easier and safer to host large numbers of mailboxes on a cluster than on a single unclustered server—The cluster provides additional redundancy so that a single failure doesn't cause downtime for all the users on the server at once. In many cases, clustering is also easier than hosting large mailbox sets on a small number of unclustered individual servers.

Don't confuse server clustering with network load balancing (NLB). Microsoft uses the term "cluster" to apply to both technologies, which is unnecessarily confusing. In this book, when I talk about NLB I'll always use the term *NLB cluster* to keep things clear.

These benefits come at a steep cost, though. First, clustering requires more servers; a two-node cluster with one active node can handle roughly the same number of users as an equivalently sized single machine, but it costs more than twice as much when you factor in the cost of shared storage. The dollars-per-user figures for clusters can look decidedly unimpressive, until you factor in the additional uptime that clusters can give you *if* they're properly designed, maintained, and administered. Speaking of which, clusters require some specialized knowledge to set up and run, as do SANs. Understanding how failover works, how to effectively troubleshoot and diagnose cluster problems, and how to manage Exchange in a clustered environment are all prerequisite skills that you need to have in your environment. Without them, you may find that clustering actually *lowers* your uptime—not the result you're looking for.

When you combine clustering and SANs, you get an extra requirement: your cluster admins have to understand SANs, and vice versa. In some large organizations, the SAN may be maintained by a group that's responsible for providing storage and doesn't know anything about Exchange, and vice versa; this situation is not necessarily a blueprint for success.

## Data Replication

Wouldn't it be great if you could have a remote server somewhere with an always-up-to-date copy of your Exchange data? That's the promise behind data replication, a loose term that I use to encompass both software- and hardware-based replication solutions. Software replication solutions typically install software on the source server that copies changes to specified files or volumes to a remote server; hardware replication copies data between a SAN controller in one site to another controller (usually the same make and model as the first) at another site.

The idea behind this approach is sound, but the implementation details can spell the difference between a solid deployment and one that causes wailing and gnashing of teeth.

Chapter 2 will discuss these implementation details and the effect they have on Exchange.

The important thing to remember is that Exchange was designed to work with local disk subsystems, and any solution that introduces too much write latency will cause problems. Replication solutions must have an adequate amount of bandwidth to efficiently replicate changes from source to target. What does "adequate" mean? It depends on the volume of changes on the source server, the replication schedule, and a host of other factors. For now, it's enough to know that the bandwidth requirement may be substantial, which represents a cost you should be aware of up front.

Software replication solutions, of course, have the advantage that they're not tied to hardware from a particular vendor; if you have a SAN from vendor A in Austin and one from vendor B in Baton Rouge, software solutions will happily replicate data between them. However, not all of these software solutions are suitable for use with Exchange, which is hardware replication's strong selling point: replication happens at the controller level, so Exchange doesn't know anything about it. Of course, you pay through the nose for that transparency.

The bigger question that you should ask when considering the cost and benefits of replication in your environment is what you'll *do* with the replicated copy of the data. It's not very interesting to just copy the data, so you'll need some way to redirect users to your recovery server in the event of a failure. Thus, you'll also need a way to fail back to your original server once it's restored; that includes both restoring users and copying back any data that changed during the outage. Both of these features are included in clustering solutions, but they're not necessarily included in all replication products. Microsoft's official position on using software replication solutions with Exchange is explained in detail on the Microsoft Web site; briefly, the company doesn't promise to support the use of Exchange data replicated by asynchronous software replication solutions, but Microsoft doesn't require that you remove the replication solution as a precondition to getting help. This is not necessarily a reason to avoid these solutions, but it does argue in favor of a careful evaluation of your chosen vendor's ability to provide immediate support if you have a problem.

### Virtualization

In the olden days of mainframe computing, virtualization allowed independent applications to run in separate logical partitions, making each application think that it had its own computer. Even though computing technology has come a long way since then, the basic idea behind virtualization remains unchanged; modern virtualization solutions such as Microsoft's Virtual Server or VMware's VMware GSX Server allow you to host multiple logical machines on a single physical server. The appeal of this technology is clear: with virtualization and a big enough server, you can operate a whole farm of Exchange servers, each one of which is really a portable container that can be easily moved to another machine for disaster recovery or load balancing. Of course, that means your service uptime is tied to *two* machines now: the virtual machine and its physical host. A hardware fault on the host, or a software fault on the host or guest machines, can torpedo the guest machine. For that reason, and because Microsoft doesn't support it, relatively few sites are using virtualization to enhance their application availability. However, as virtualization technology improves, and as Microsoft moves to make Exchange work better with its own Virtual Server product, I expect to see it used more often as a first-line mechanism.

### Business Continuance

Business continuance (BC) is an often-misunderstood *process*; it's not a single technology or procedure; it's a set of processes and procedures designed to help you keep operating during an outage and the recovery period that follows. From that perspective, it encompasses some aspects of disaster recovery and some of service provisioning, since one important BC goal is to quickly restore some level of service to critical employees while an outage is still ongoing. When most administrators think of BC, they think of alternative recovery sites and large-scale disaster recovery. These are both important in enterprise BC planning, but it's possible to get by with smaller expenditures. The specifics of a BC implementation will vary according to the business, its requirements, and its budget; there are several ways to provide BC services, and I'll discuss them in more detail throughout the remainder of the book. However, all of them are relatively expensive, which is to be expected if you're talking about solutions that essentially involve duplicating some portion of your messaging operations and infrastructure.

### Fault Tolerance

If you're operating a nuclear power plant, the New York Stock Exchange, or an air traffic control center, you might need fault-tolerant systems. In the world of people who design and build high-availability systems, the words "fault-tolerant" have special meaning; they refer to systems that can tolerate hardware and software faults while remaining in normal operation and allowing the fault to be repaired by replacing hardware or reloading software.

Many of the features first introduced in fault-tolerant systems from companies such as Tandem (for example, hot-swappable disks, redundant power supplies, and uninterruptible power supplies) are now common in mainstream hardware. However, other fault tolerance features, such as having multiple redundant CPUs that run the same code in lockstep and "vote" on the correctness of the results, are still the province of high-end fault-tolerant systems. The complexity and expense of these computers, and the specialized software that runs on them, mean that they're not appropriate for the vast majority of businesses that depend on messaging.

## Assessing Your Exchange Environment

So far, this chapter has focused mostly on downtime. The next logical step is determining how to have *less* downtime in your environment by assessing your Exchange environment and the business needs it serves. This topic is what the rest of this book is about, but before talking about the detailed technical solutions you can employ, you need a framework for deciding which technologies make sense for your particular environment. In other words, what business justifications exist for an increased level of availability? What should you do, in what order, to bring up your availability to the desired level?

Of course, it's likely that since you're reading this book, you're in charge of actually running your Exchange environment, so you might wonder why you should read a section that talks about cost justification and return on investment (ROI). The answer is simple: it's difficult to accurately estimate how much money you should spend on increasing your availability unless you know two things: how much downtime is costing you and how much downtime you have. This data will guide you in defining appropriate service levels, taking into account any existing commitments you have to providing a particular level of uptime.

### *Trading Money for Uptime*

How do you decide how much money is reasonable to spend on availability, disaster recovery, and business continuance? In the time-honored manner of insurance salesmen and gangsters, just ask yourself what happens if you *don't* spend the money! More seriously, every organization will suffer to some degree during a messaging outage; the degree of impact will depend on what you do and how integrated messaging is with your business practices.

In most cases, the actual loss caused by an outage is related to customer contact, such as an inability to accept orders. This cost is relatively easy to quantify if you know what the volume of customer orders per hour is, or if you have some other metric (such as the number of customer contacts) to use in its place. For organizations that depend more heavily on email as a communications and collaboration tool, loss of messaging service may carry a heavier, but harder to quantify, price. For example, many law firms are totally dependant on email and calendaring to get their work done; without it, highly paid attorneys and paralegals may not be able to do billable work, resulting in immediately identifiable losses.

How do you come up with a number? Stratus Computer Corporation (maker of fault-tolerant systems) has an Excel spreadsheet at http://www.stratus.com/vsc/calc.htm that can give you a decent first-order estimate. To use it, you need to know:

- When your peak operating hours are, and how many total hours, days, and weeks per year your business is operating
- How many users are active during your peak operating times, and the average annual salary for all users considered as a group
- What percentage of your users' time can't be diverted to some other productive task
- How much revenue per hour is dependent on your messaging system; alternatively, you can specify the number of hourly transactions (sales, orders, tickets booked, and so on) and an average value per transaction
- The number of hours, and hourly cost, for IT staff involved in fixing outages
- Any penalties or other costs associated with outages

After you plug in these factoids, the Stratus calculator will tell you what it thinks your hourly cost for downtime is. Of course, you can always derive your own cost of downtime based on these same metrics, or others that make sense for what you're doing.

### *Technical Assessment*

Armed with an idea of what downtime actually costs you, the next step is to figure out how much downtime you're currently experiencing. This determination is easy if you have a small number of centrally managed servers; it gets more difficult if you have a more distributed infrastructure. Even with a small number of servers, it can be challenging to come up with accurate statistics for downtime when you consider that dismounting one database on one server counts as downtime for the users whose database was dismounted.

Start with the simplest measurement: in the last year, did you have any total outages in which no users could send or receive mail? This measurement is easy to figure out, and it gives you a good start on finer-grained measurements. If you have management or monitoring tools that give you more visibility into the actual amount of downtime you've experienced, so much the better.

Next, try to figure out the amount of downtime that was (or should have been) planned downtime for maintenance. It's fair to exclude that from the total, but only if your service level agreements (SLAs) make explicit provision for planned downtime. If they don't, then you should plan to revise them so that they do.

The remaining amount of downtime is the amount chargeable as actual outages. For each outage, you should be able to identify a cause. If you can't, that should be a warning that you need to strengthen your monitoring procedures so that every outage can be linked back to a cause—we'll talk more about that in future chapters. Assuming that you can assign a cause to each outage, then you're prepared to start deciding where to allocate your attention and funding.

Part of this decision-making process is identifying any weak spots in your current configuration so that you can prioritize fixing the worst deficiencies first. To this end, you should download and run the Microsoft Exchange Best Practices Analyzer (ExBPA), which is available from Microsoft's Web site. ExBPA will give you a great deal of information about the configuration of your Exchange servers, and it will highlight any areas that deviate from Microsoft's configuration recommendations. Because ExBPA is free and doesn't make any configuration changes, you can easily run it to get an idea of potential problem areas in conjunction with your other, more detailed, analyses.

### *SLAs: Explicit vs. Implicit*

The final part of your assessment should revolve around deciding what service levels are acceptable for your messaging system. SLAs are the catch-all term used to refer to explicit or implicit promises or expectations about the uptime of a business-related system.

Everyone has an implicit SLA, even if they don't necessarily realize it. Not long ago, I worked with a large video game company that wanted to improve their messaging stability and uptime to four nines or better. When I asked what the business justification for doing so was, the IT director told me that the company's database servers normally hit 99.99 percent planned uptime, so that was the executive-level expectation for messaging. The success of their database operations set an implicit SLA, in that the C-level staff all expected messaging to be just as available as the database operations—never mind that messaging is distributed across four continents, eight countries, and more than a dozen physical sites! Implicit SLAs are hard to meet, because you often don't know what they are until after you've broken them!

Explicit SLAs, in contrast, set out *exactly* what expectations are. You can think of them as formal contracts; in fact, telecommunications companies often write SLAs into their contracts so that if they exceed a certain amount of downtime they owe you money. More often, internal SLAs promise that—in return for adequate funding—the IT operation will deliver a certain level of performance, availability, security, and so on.

If you already have explicit SLAs, good! That means that you have a commitment to work from as you plan how to meet, or exceed, your availability goals. Review them carefully to see what they tell you about underlying assumptions, then review your actual uptime to determine whether you're currently meeting the requirements you already have.

If you don't have an explicit SLA in place, you should create one based on your best estimates of what kind of uptime will meet your business needs. Begin with whatever data you have on your implicit SLAs; those make a good starting point for an open discussion on what kinds of service levels you *should* be providing. Be sure that your proposed SLA includes adequate time for planned maintenance!

## Summary

The technology of protecting Exchange systems against unwanted downtime is interesting, but before you can effectively use it, it's important that you gain an understanding of what downtime *is*. This chapter explored the difference between planned and unplanned downtime, some of the major causes of downtime, the general technologies that exist for increasing availability and resiliency, and ways to assess the current state of your Exchange environment to see where your remediation efforts should begin. Chapter 2 will examine the basic technologies that can be used for disaster recovery.