Realtime publishers

The Definitive Guide™ To

Windows Application and Server Backup 2.0

sponsored by

Don Jones

Copyright Statement

Chapter 6: SharePoint Server Backups

Microsoft's SharePoint Server has probably had the most variety in its backup and restore solutions. The first version of the product was essentially a modified version of Exchange Server, and used the same database engine that Exchange did at the time. Today, SharePoint Server uses multiple databases to store its content, configuration, search catalogs, and more—and even stores some critical files as simple disk files. All that data stored in different places helps make SharePoint Server one of the most difficult Microsoft server products to work with in terms of business continuity and disaster recovery. It becomes even more complex when you start dealing with SharePoint Server *farms*—collections of servers designed to serve up the same content for load-balancing purposes. Is it even possible to move beyond the Backup 1.0 mindset and start using Backup 2.0 when it comes to SharePoint?

Native Solutions

Microsoft defines three levels of data recovery for SharePoint Server:

- **Content recovery** is when you recover one or more items using a Recycle Bin or retrieve a previous version of the items from the content database. This relies on functionality within SharePoint itself and is accessible to end users.
- **Site recovery** is when you recover an entire SharePoint Server *site,* or Web site. This is the type of recovery most administrators are concerned with.
- **Disaster recovery** typically involves site recovery to new hardware.

Content recovery doesn't affect anything but the SharePoint content store; users utilize the SharePoint user interface (UI) to access prior versions of a file or to recover deleted files from the in-product Recycle Bin. Other forms of recovery, however, deal with the search catalogs, site configuration, and other data stores. Natively, SharePoint Server supports a dizzying array of backup and recovery options, all of which essentially overlap to provide full coverage of the product's data. I'll cover some of the major native options in the next few sections.

Versioning

Versioning is designed to provide single-item recovery for past versions of an item. When a user overwrites an existing document, SharePoint retains the old version. Users can always retrieve those old versions, going as far back as SharePoint has retained. Versioning must be enabled by an administrator, and the administrator can select two options:

- **Major versions**. Each new version of the document is given a sequential number, such as 1, 2, and 3. All users have access to all versions, and you can specify how many previous versions are retained.
- **Major and minor versions**. This allows for minor versions (1.1, 1.2, and so on) of documents, although only *major* versions (those ending in 0) can actually be published; everything else is considered a draft revision. You can choose how many major and minor versions you will keep.

Versioning is an effective recovery tool for end users—to a point. It's not practical to store an infinite number of versions, so at some point, you may still have to go to a tape-based backup in order to retrieve a past version that is older than what's retained in the SharePoint database. Versioning also assumes that SharePoint is up and running, meaning it isn't effective for disaster recovery or other scenarios where the entire server is lost.

Recycle Bin

SharePoint includes a "two stage" Recycle Bin. The first "stage" exists at the end-user level, providing a simple "un-delete" feature that lets users recover accidentally-deleted files and other items from within the site. The second "stage" is accessible to administrators, who also have access to all the site's first-stage Recycle Bin items. When a user "empties" their first-stage Recycle Bin, the data is retained in the second-stage Recycle Bin for recovery by an administrator.

Items are also removed completely after 30 days (by default) in the Recycle Bin system. After that time period elapses, it's back to backup tapes to recover older items. Items are also removed from the second-stage Recycle Bin when it reaches its storage quota; the oldest items are permanently deleted to make room for new items. As with versioning, the Recycle Bin depends on SharePoint being up and running, so it isn't intended as a disaster recovery or whole-server recovery tool.

SharePoint Designer and STSADM Import/Export

Office SharePoint Designer has the ability to back up and restore individual SharePoint sites and site collections, down to the individual file level. The tool utilizes the STSADM tool to export or import the site; prior to SharePoint Server 2007 Service Pack 2 (SP2), backups were limited to 25MB—which isn't much. After SP2, backups are limited to 2GB, which is a lot of space but still might not be enough to completely back up a very large, busy SharePoint site.

The backups created in this fashion are actually content migration packages; the only way to utilize these backups is to restore the entire site—often to an offline server that is used specifically for that purpose. Backups do not include Recycle Bin files, so anything that was deleted when the backup was made will not be included in the backup. The backups also do not include workflow definitions, alerts, or site-collection properties—meaning these backups do *not* contain *everything* that defines your SharePoint site.

You can also use the STSADM tool by itself to export a site (create a backup) or import a site (restore from backup). The tool has the same limitations when used by itself or from the SharePoint Designer.

There are some other weaknesses of STSADM, including the fact that, although it can back up the SharePoint configuration database, it can only restore the configuration to a server of the same name—making it useless for disaster recovery. Microsoft's guidance document for SharePoint backup and recovery (http://technet.microsoft.com/en-us/library/cc262129.aspx) notes some other disadvantages:

- Cannot back up directly to tape—backup location must be a UNC path
- Does not provide automatic deletion of old backup files
- As part of a farm backup, can back up the configuration database and the Central Administration content database but will not restore them
- Does not back up any custom solution files in the Inetpub or Office 12 hive (Program Files\Common Files\Microsoft Shared\Web server extensions\12)
- Does not back up alternate access mappings (AAM)
- Does not back up Internet Information Services (IIS) settings including host headers, dedicated IP addresses, and Secure Sockets Layer (SSL) certificates
- Site collection backups affect performance and can cause access errors and should only be used when the site collection is locked; site collection backups can be slow when working with collections larger than 12 to 15GB, so Microsoft recommends that you use farm backups if you are working with collections larger than 15 GB

Content databases larger than 100GB—which isn't really that much in today's world—aren't suitable for STSADM, and Microsoft actually recommends you go buy something from someone else to handle your backups.

SQL Server Backup

Because SharePoint Server uses SQL Server as its main data store, SQL Server's own backup tools can be used to create partial backups of a SharePoint site. I say *partial* because not *all* of the SharePoint site data is inside SQL Server.

You can also create a SQL Server *database snapshot* of the SharePoint site. This is a read-only version of the database as it exists at the time of the snapshot. These snapshots shouldn't, however, be considered real backups, although they do provide a means to retrieve data from the database as it exists at the time of the snapshot—such as recovering individual items. Doing so requires a SharePoint site to be created that uses the database snapshot, and that site will be read-only; obviously, SharePoint Server needs to be functional in order to do that. There are other disadvantages—again, from Microsoft's guidance document:

- Does not include front-end Web server custom solutions
- Can back up the configuration database and Central Administration content database but restoring is not supported
- Does not back up IIS settings set outside of Office SharePoint Server, including host headers, dedicated IP addresses and SSL certificates
- If using Search, you must re-crawl after restoring content because indexes are not backed up in SQL Server
- Should not be used to back up the Search database because it cannot be synchronized with the Search index
- You must manually reattach your databases to the Web applications after a recovery

Why do these native tools support backing up the configuration database but not restoring it? What would be the point, exactly?

Central Administration

The central SharePoint administrative interface provides basic backup and recovery capabilities. It's easy to use, and—brace yourself—can automatically handle backups that run more than 17 hours. I know, the very thought of a 17-hour backup, let alone something that runs longer, is so mired in the Backup 1.0 mindset that I don't even want to think about it.

Central Administration does, however, have a lengthy list of disadvantages (from Microsoft's guidance document):

- Does not provide scheduling functionality
- Does not provide automatic deletion of old backup files
- Cannot back up directly to tape; backup location must be a UNC path
- As part of a farm backup, can back up the configuration database and the Central Administration content database but will not restore them
- Does not back up any configuration changes or custom solution files in the Inetpub or Office 12 hive (Program Files\Common Files\Microsoft Shared\Web server extensions\12)
- Does not back up any customizations made to the Web.config file

- Does not back up AAM
- Does not back up IIS settings, including host headers, dedicated IP addresses, and SSL certificates
- If a backup or recovery job is not successful, the unsuccessful job must be manually deleted from the Timer job list on the Backup and Restore Status page; in Central Administration, click **Operations**, then click **Backup and Restore Job Status**—if the failed job is not deleted manually, subsequent backup or recovery jobs fail

So, it has to be used manually, doesn't back up everything, can't back up to tape, and has to be manually monitored and maintained. Wonderful. That's not even Backup 1.0; it's more like Backup 0.5!

Problems and Challenges

The biggest problem with the native solutions is that—well, frankly, they're all over the place. None of them are real, tried-and-true backup solutions in the traditional sense of the word. None of the native solutions I've discussed provide for disaster recovery, for example, which is important even in the Backup 1.0 mindset.

Disaster recovery, in fact, can be incredibly complicated with SharePoint. You have the content database, customizations, SharePoint-level configuration settings, binary files (SharePoint Server's own files), IIS-level configuration settings, and finally the binary files of the Windows operating system (OS) itself. *Nothing* in the native toolset can back up *all* that information.

Microsoft's recommendation for disaster recovery is to:

- Use Windows Server Backup to back up the binary files—although in a full disaster recovery scenario, Microsoft recommends *reinstalling* the OS, SharePoint, and SQL Server, which will take several hours to complete.
- Document—write down, that is, not actually back up—the IIS configuration settings. You're supposed to manually re-create the correct configuration in a disaster recovery scenario.
- Document—again, write down, not actually back up—the SharePoint configuration settings. Although it's possible to back up these and restore them, doing so is dangerous because they need to be perfectly synchronized with the other data. Failure to maintain that synchronization, which is difficult to do manually, will result in random site errors and force you to re-create the SharePoint site from scratch.
- Package any customizations as solutions so that they don't have to be backed up but can instead be re-installed if you need to rebuild the site.
- Back up the SQL Server databases to protect the site content. You can use SQL Server tools for this, except in the case of the Search database, which *may not* be backed up using SQL Server tools. You have to use SharePoint tools to back up that.

Honestly, you've got to be kidding. SharePoint appears to be a product that simply *does not like to be backed up.* Sure, it's nice to have great functionality such as versioning and a Recycle Bin for single-item recovery by end users, but what about a complete disaster? How, exactly, are you supposed to prepare for a whole-server failure? And for that matter, how are you supposed to go about recovering data that's too old to be in a Recycle Bin or in the versioning system? What if you never turned on versioning and need to recover an old file?

Part of the challenge, of course, is that SharePoint is a huge and complex product (Figure 6.1 illustrates the various independent components) that relies on numerous technologies outside the control of the SharePoint development team at Microsoft. It relies on specific IIS configuration settings, and IIS has never made it especially easy to back up those configurations (although, as of IIS 7, it's gotten easier—Web site settings are in a simple web.config file that can be backed up like any other file). SharePoint sites often have customized files, which reside on the server's file system. SharePoint data lives in SQL Server, which has its own backup and recovery model. Worse, much of SharePoint's configuration data—as I've described—relies on being in exact sync with the data stores, so you pretty much have to back up everything at once. From a backup and recovery standpoint, SharePoint just isn't well thought out, unfortunately.

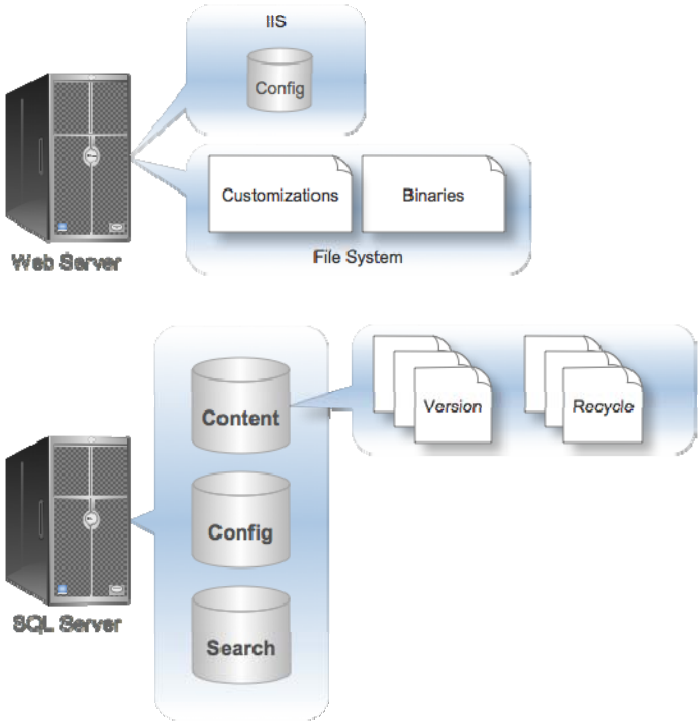

Figure 6.1: The many components of SharePoint Server.

So how does Backup 1.0 work in a SharePoint environment?

In the Old Days

The weaknesses of Backup 1.0 are really brought into sharp relief by a complex product like SharePoint Server. Microsoft's recommendation? Buy a backup solution, such as the one Microsoft will be happy to sell you. Many backup solutions support SharePoint Server, but all do so from a very Backup 1.0 perspective: point-in-time snapshots, lengthy backup windows, and even longer recovery times.

Backup Techniques

For standard backups, you have two choices: Use the mishmash of tools that are native to SharePoint or available for free, or buy something that can back up the entire SharePoint infrastructure in one go. Either way, large SharePoint installations can have incredibly long backup times—the fact that the native tools have specific features to accommodate backups longer than 17 hours should tell you something.

I know a *lot* of SharePoint administrators who run their SharePoint Server installations—SQL Server and all—in a virtual machine simply because that allows them to grab copies of the virtual machine's virtual disk file, thus backing up the OS and all the SharePoint bits in one piece. Figure 6.2 illustrates the basic idea.

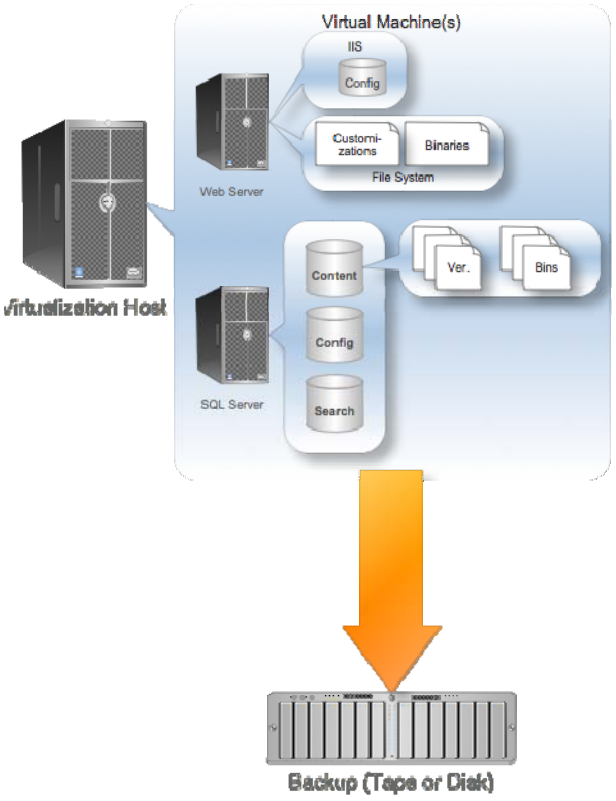

Figure 6.2: Easier backups with virtualized SharePoint.

The problem with this approach is that it's tough to actually use those backups for anything *except* disaster recovery. You're stuck restoring the *entire server,* and if you only needed to recover one file—well, that's definitely a bit of overkill.

Other Backup 1.0-style solutions try to duplicate the approach of the native tools but consolidate everything in one tool for you. In other words, these solutions dig into SharePoint and grab the IIS configuration, the content, the SharePoint configuration, and so on—but rather than giving you eight tools to do it, they do it all at once. That can be beneficial, but it's still a point-in-time snapshot, and you've always got data at risk. Performing recovery from that kind of backup can sometimes be tricky, too, depending on how the backup solution is built.

By the way, I don't want to imply that Microsoft hasn't thought about all these complexities. They have; one of the "complete" backup and recovery solutions that you can buy is made by Microsoft.

Restore Scenarios

Recovery scenarios using the native toolset are just as complicated as performing the actual backup using the native toolset. In fact, as I was researching this chapter, I commented on Facebook that I couldn't believe how complex the native toolset was; one of my friends responded:

> "It's more like they never expect the systems to ever fail & thus need a restore! HA! I am so happy to be off of the SharePoint team!"

Ouch. But I couldn't agree more: Natively, SharePoint *does* assume that the system will never fail, and that Recycle Bins and versioning will provide all the recovery capability you'll ever need. If you do experience a complete system failure, the official recommendation is to essentially start from scratch and reinstall everything, then restore your content database from a standard SQL Server backup.

Of course, Microsoft isn't that naïve—the company absolutely recognizes the need for full system recovery, and that's why they make an add-on product that will do it for you. However, once you've decided that you have to spend money to protect SharePoint—and, frankly, you do—you should start looking beyond Microsoft and compare their offerings to the other solutions out there.

Those other solutions—including the one Microsoft will sell you—can sometimes do a better job at handling the two recovery scenarios you'll run into: single-item recovery and disaster recovery. I'll get into disaster recovery next; from a single-item recovery perspective, most solutions work by pulling the data out of SharePoint and storing the backed-up data in a way that *they* can also access. It's an approach similar to the way some solutions back up Exchange Server: Rather than attempting to grab the entire database in one chunk, they grab each individual content item. That way, you don't need a copy of SharePoint to access individual items from the backup—you just use the backup software itself.

Individual Item Backup: Great for Deduplication

Invariably, SharePoint sites wind up with a lot of duplicated data copies of files kept in multiple places within the site, for example. A solution that accesses individual items for backup purposes can compare them with each other and eliminate duplicates from the backup, making the backups smaller and saving space. *So much* space can be saved in a typical SharePoint installation that "deduplication" has become a *major* selling point for SharePoint backup and recovery solutions.

Other solutions take a slightly different approach to how they make their backup, but the practical upshot is this: You *want* a solution that will allow you to access one or more individual items *without* having to restore the entire database to a live SharePoint Server. There is actually a whole market of tools that can add this capability to other backup solutions. For example, tools exist that can open and browse the backup files made by Microsoft's own extra-cost backup solution, enabling you to browse the SharePoint repositories without actually performing a database restore. Figure 6.3 shows one such solution, which is a free tool available at http://www.appassure.com/applications/docretriever/. It can connect to SQL Server backups, STSADM exports, and other backup snapshots; let you browse the database; and retrieve individual items from it.

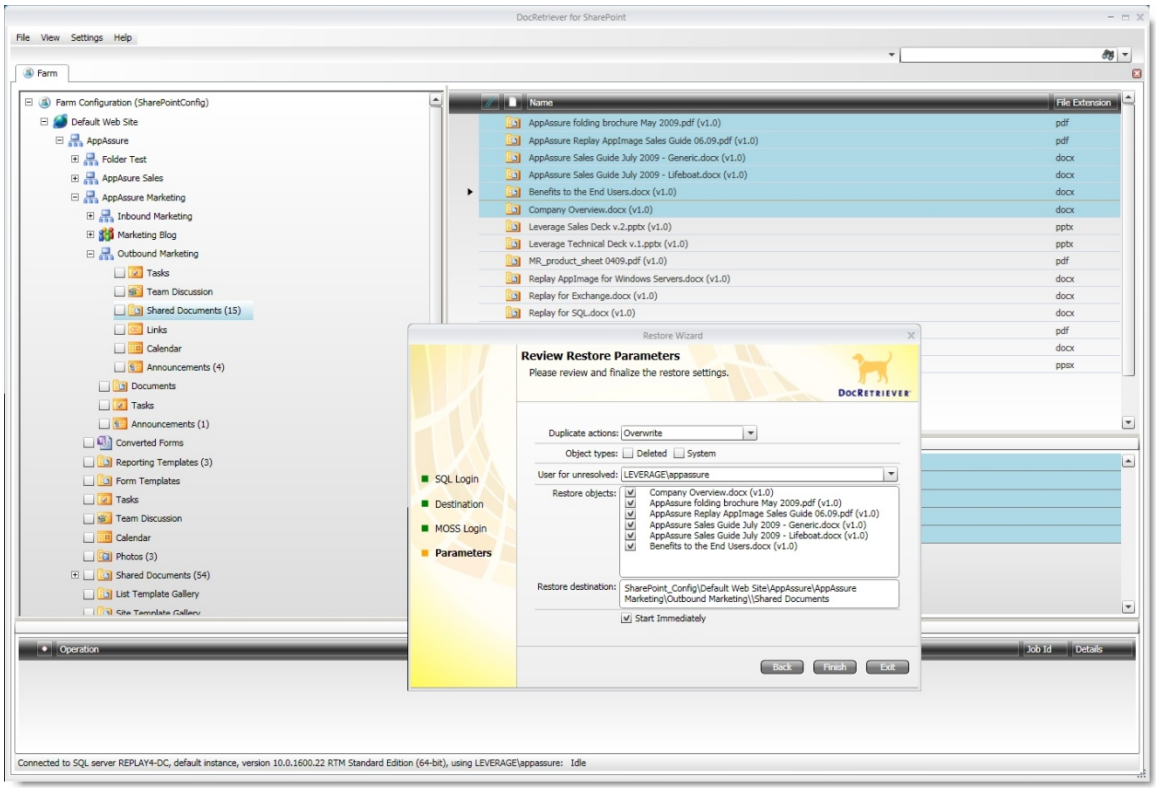

Figure 6.3: DocRetriever can recover individual items from a SharePoint backup file.

Disaster Recovery

As I've already said, the official disaster recovery policy for SharePoint is to reinstall, reconfigure, and then restore your content database—a process that will, in most cases, take a day or two *at least* (including time to apply patches and service packs to the OS and application software)—and that's assuming you've done everything by the book, packaged any customizations as deployable solutions, and so on. If you haven't done those things…well, sorry. You're in for a long recovery project.

I've also mentioned virtualization as a way to make disaster recovery easier. By backing up a virtual disk file, you can restore that file in a single operation—thus restoring your OS, SharePoint binary files, databases, configurations, and everything all in a single operation. Of course, you may be limited in how often you can effectively grab a backup of a virtual disk file: To get a consistent, reliable copy of the file, you're going to need to shut down the virtual machine. A large SharePoint installation may use virtual disk files that are dozens of gigabytes in size, which may take some time to copy—although you can make a disk-to-disk copy, then back up the *copy* to disk while you restart the virtual machine.

> **For Example**
>
> I have one customer whose SharePoint site includes about 110GB of files and data; their virtual machine's virtual disk files total about 146GB. Even on a fast storage area network (SAN), it takes a few hours to make a disk-to-disk copy of all the virtual disk files, and it takes *several* hours to back the copy up to tape for offsite storage.

If you're using an add-on backup solution, odds are that it provides a disaster recovery path. Some require that, at a minimum, you reinstall Windows, and may require that you reinstall SQL Server and SharePoint also; others back up the entire server and can restore the entire server. From a disaster recovery perspective, I prefer solutions that can restore the complete server to bare metal because that's the easiest, most foolproof way to get the server up and running again as quickly as possible.

Backup Management

Managing backups in a SharePoint Server world can be complicated—a bit more so than the usual backup-tape-shuffle that we're all familiar with. Because the various bits of SharePoint—configurations, content, databases, files, and so forth—are all so tightly interconnected, you have to make sure that all the various backups match up. In other words, you need to fit everything from a single backup onto a single set of backup tapes.

Because SharePoint backups can be so time consuming, administrators tend to rely on techniques such as differential and incremental backups, which allow you to make more frequent backups without such lengthy backup windows. However, all those incrementals and differentials also complicate backup management and make recovery processes longer and more complex.

I remember one particularly unrewarding experience with SharePoint, where we had a complete backup from about a month back and incremental backups since then. We were using a third-party backup solution, and we made an incremental backup every night. One day, the server died and corrupted a lot of disk files, so we needed to restore everything. That's when we found out that the incremental backup of the SharePoint configuration database—a tiny, tiny little file—from about a week back was corrupted. That meant all the backups of that same database since then were useless because they all relied on having a continuous sequence of incremental backups. Because the configuration is so closely tied to the content and everything else, we basically had to stop restoring data at that point—meaning we lost over a week of data, even though we had more recent backups for the actual content. We tried several times to extract the data in various ways; ultimately, we wound up using a third-party solution to connect to the content database backup, and manually extracted the individual items that had changed. *Not* fun, and it took almost 2 weeks to get everything almost back to normal. That's Backup 1.0 for you.

Rethinking SharePoint Server Backups: A Wish List

Can Backup 2.0 make SharePoint backups any less complicated? It seems like Backup 2.0 may have met its match: This is the first time I've looked at a server product that has its own binaries, relies on IIS, relies on SQL Server, and stores files and data in a half-dozen different places. The Backup 2.0 concept has its work cut out for it.

New and Better Techniques

I've mentioned that some administrators virtualize their SharePoint installations simply to make backups less complicated—and it turns out that those administrators are on the right path. Not with virtualization per se, but rather with the idea that backing up the individual *components* of SharePoint is overly-complex; the right idea is to back up the *entire server*, grabbing SharePoint's data along with everything else.

Take a look at Figure 6.4. I've re-drawn Figure 6.1 to simplify it a bit and to illustrate that although SharePoint has many components, *all* of them store their data on the server's file system. Disks are, after all, the only persistent form of storage in a typical server.

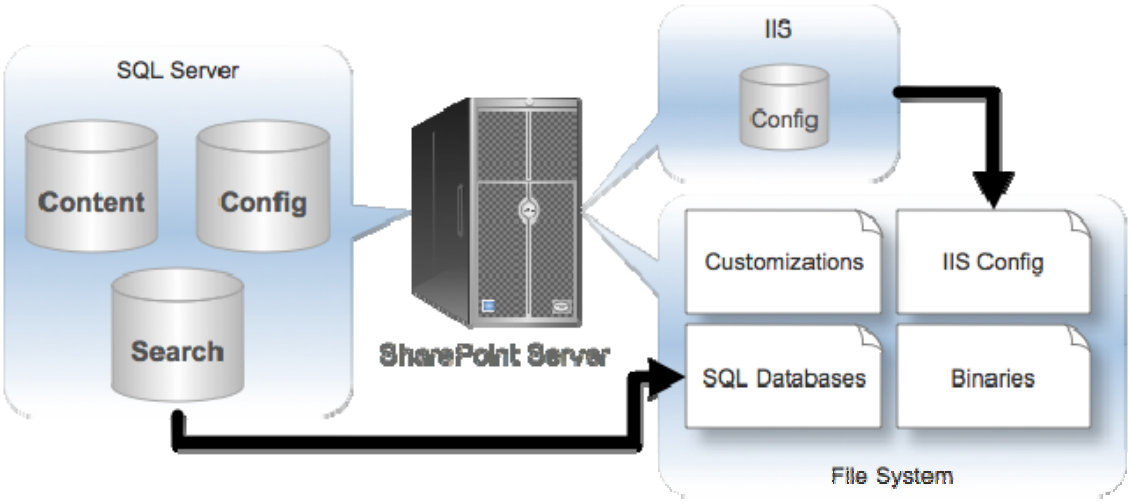

Figure 6.4: Everything in SharePoint ends up on disk eventually.

The trick to making SharePoint backups easier is to *forget about SharePoint*. Ignore the databases, ignore the files, ignore the configurations, ignore it all. Instead, simply back up *every block of disk space.* As data is changed, that data is written to disk in the form of changed disk blocks—have an agent on the server grab those changes and send them off to a backup repository.

Those backed-up disk blocks can also be time-stamped, meaning the Backup 2.0 solution can keep track of which changes came from what point in time. That immediately solves the problem with keeping the configurations synchronized with the content; so long as you're grabbing all the disk blocks up to a specific point in time, you'll have a consistent SharePoint restore. That also means you can choose to restore *up to any point in time,* effectively "rolling back" SharePoint to some earlier point in time, if needed. Figure 6.5 shows how an agent might capture the changed disk blocks and send them to a time-stamped repository.

The beauty of the Backup 2.0 approach is that the backup solution doesn't really need to know or care what's running on the server, or how complicated it is. Everything that matters to us as human beings will end up on disk; so long as we capture every single changed disk block, we've got a complete backup. Because a disk block is tiny—no more than 64 kilobytes, and as small as 4 kilobytes—backing up a single disk block isn't very time consuming or intensive. Streaming those changes over the network—using compression, of course—is entirely possible, and gets all the backed-up data to a safe place almost in real-time.

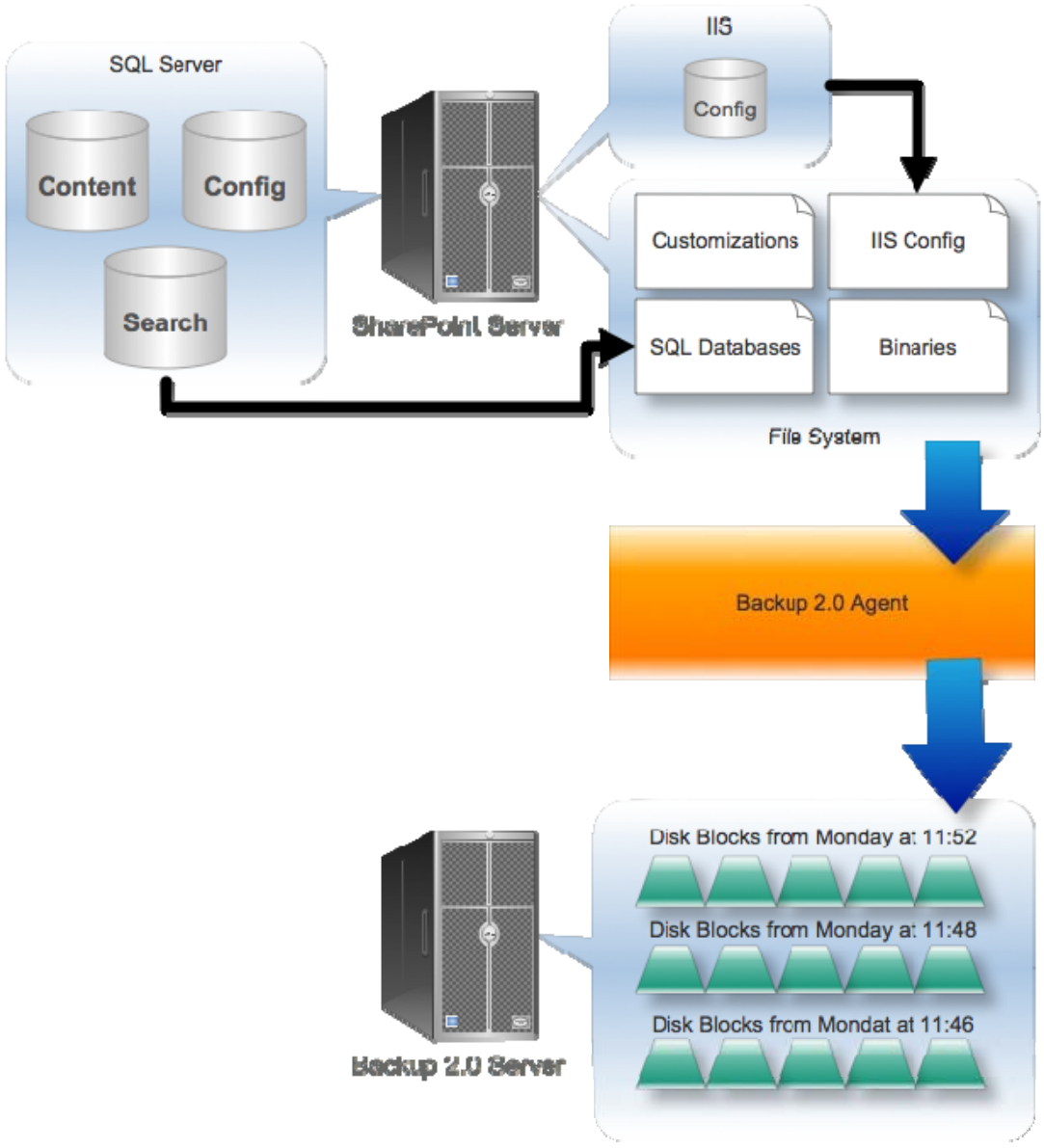

Figure 6.5: Streaming disk blocks to the backup server.

Deduplication of data is also possible, meaning backups can be smaller, take up less space, and even require a bit less time to restore.

Better Restore Scenarios

As I said before, you really get into two recovery scenarios with SharePoint: needing to recover a single item and needing to recover an entire server. The latter is what I call disaster recovery, and I'll get to it next. For now, does Backup 2.0 improve single-item recovery?

Absolutely. Remember, because those backed-up disk blocks are time-stamped, you can recover the entire server to a specific point in time—*or* you can *expose the data stores* as they existed at a certain point in time. This is a concept I've mentioned before, but it's particularly applicable to SharePoint, so let me run through an example.

Let's say your goal is to retrieve a single document from SharePoint, as that document existed at noon on the previous Monday. The document lives in the content database, which is a SQL Server database. SQL Server databases are files that sit on the file system; typically, a certain amount of disk space is allocated to the database, whether the database actually contains that much data or not. As the allocated space is filled, the database file is expanded to make room for more data. So let's say that this particular database occupies disk blocks 10,000 through 790,000 (for this example, we live in a perfect world where disk fragmentation doesn't exist, so the database occupies a contiguous sequence of disk blocks). Figure 6.6 shows a portion of this disk space—blocks 11,100 through 11,123.

Figure 6.6: Disk blocks on the file system.

When we start our Backup 2.0-style backup, we take a snapshot of every disk block on the server as our base, or starting point. Then, we simply capture disk blocks as they change. Figure 6.7 shows a partial example: the base snapshot for our range of disk blocks as well as changed disk blocks captured from three time periods.

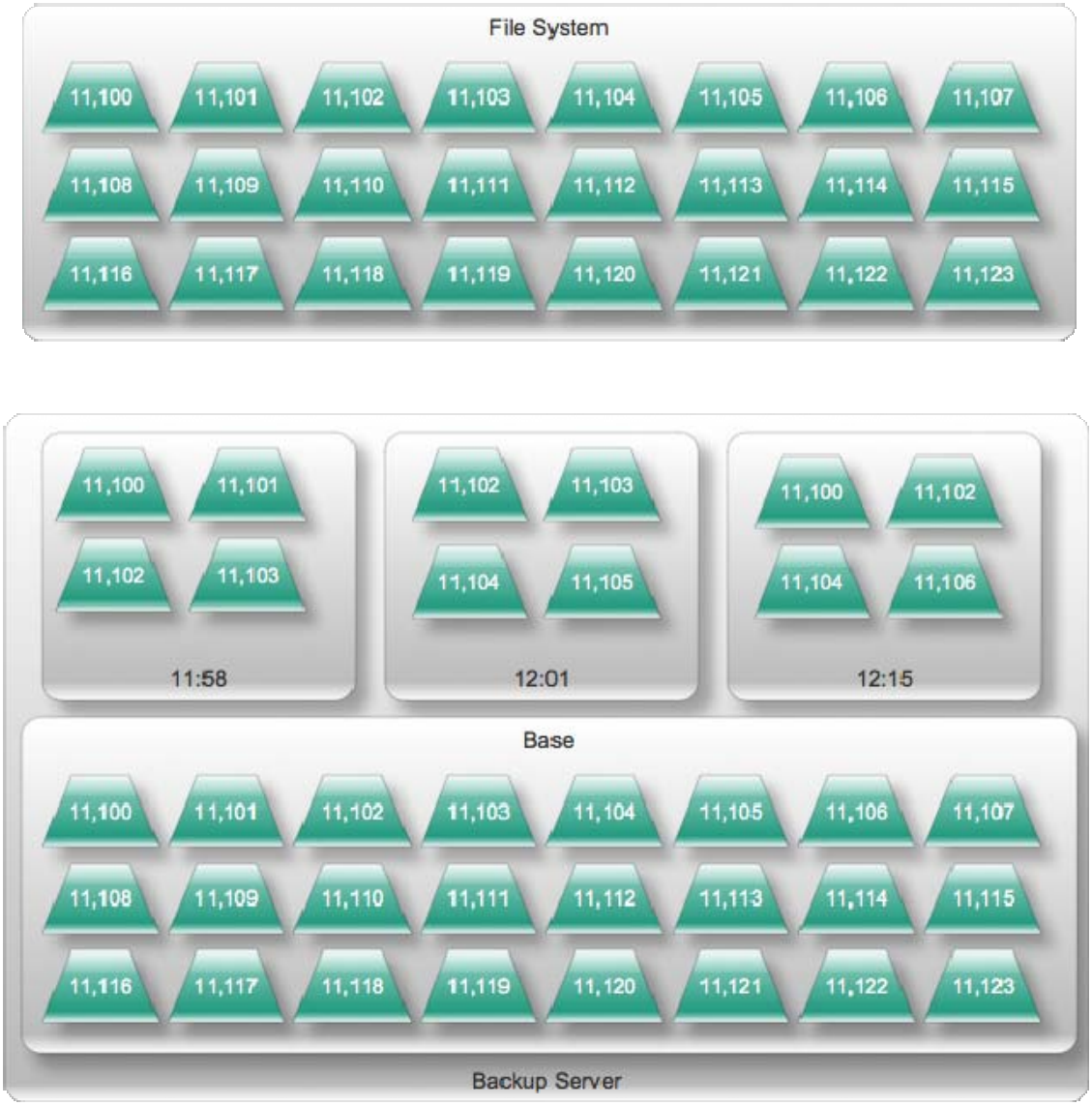

Figure 6.7: Backing up disk blocks from the file system.

When the time comes to perform our restore, we need to re-construct the file system as it was at, say, 12:02. So the backup solution goes and starts with the base snapshot, then substitutes any disk blocks that have changed between the time that snapshot was taken and the recovery time we specified. In the event that a disk block has changed multiple times—11,102 is a good example—we use only the latest version, up to the recovery time specified. Let's say that we're only concerned with disk blocks 11,100 through 11,107 because those hold the data for the file we want to recover. Figure 6.8 shows how the backup solution would apply blocks to re-create the server's file system as of 12:02.

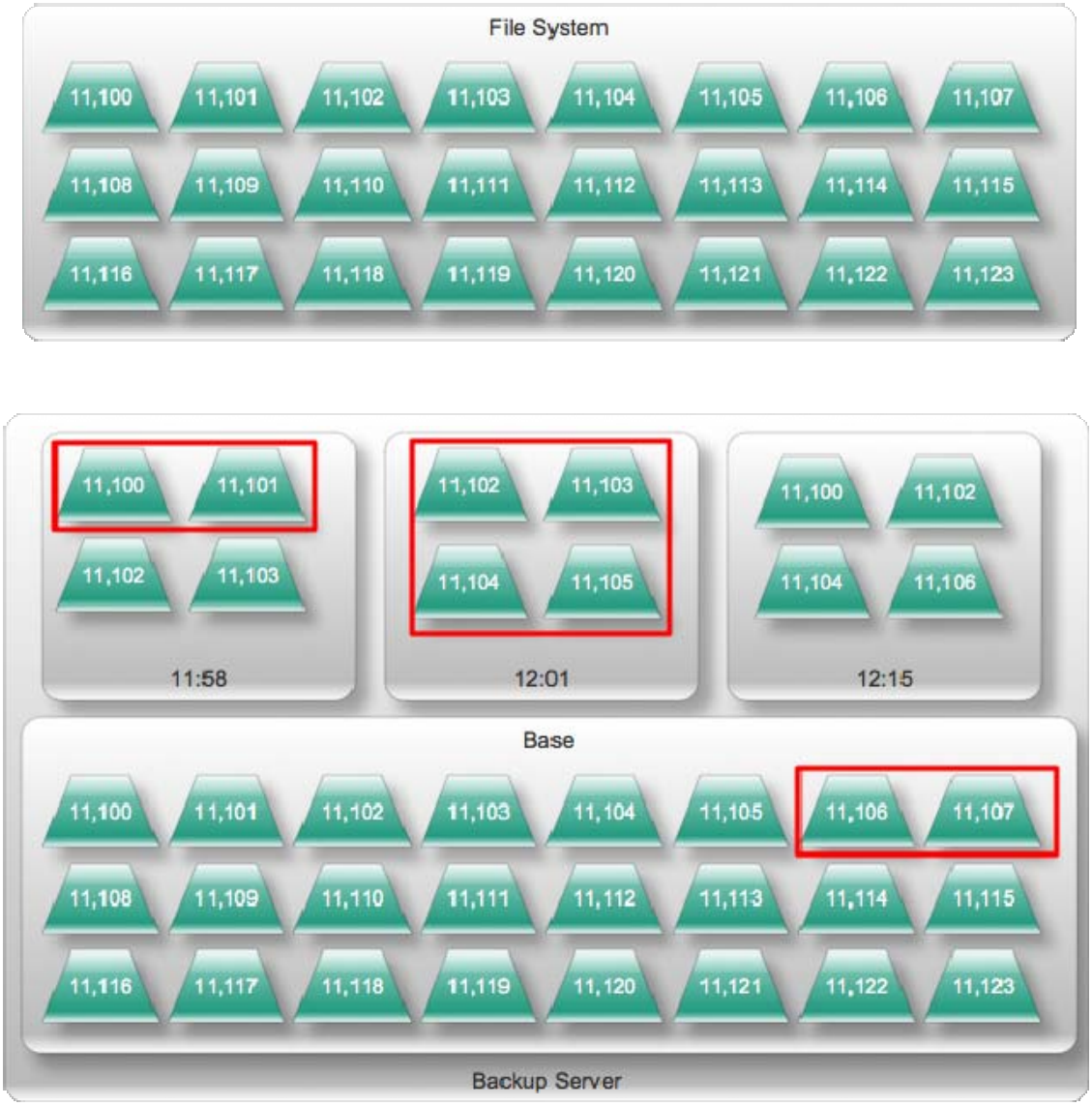

Figure 6.8: Recovering to a specific point in time.

Once the backup solution has re-constructed this point-in-time file system, we have two choices. We could simply restore the entire server to that point in time—perhaps recovering to a virtual machine so that we could use SharePoint itself to actually retrieve the file. Or, the backup solution might be able to expose this point-in-time view of the file system as a mountable image, meaning we could browse its file system, attach the database to a SQL Server instance, or even use a recovery tool (like the free one I mentioned earlier) that can attach to the SQL Server database without needing an actual running instance of the SQL Server software.

This is a *much* better recovery scenario: It's easy, pretty intuitive, and doesn't require a lot of work. A backup solution might be able to construct this point-in-time snapshot almost on the fly, meaning we could get in and browse specific points in time whenever we needed to.

Better Disaster Recovery

Disaster recovery is easier under Backup 2.0, as well. Remember our mission statement for Backup 2.0:

> Backups should prevent us from losing any data or losing any work, and ensure that we always have access to our data with as little downtime as possible.

Because we're streaming those disk blocks in near real-time, we won't lose very much data at all—a few seconds' worth, perhaps. We can access our backed-up data almost instantly, as I've described, and we can restore an entire point-in-time image anytime we need to perform disaster recovery.

SharePoint is pretty picky about its configuration. If you set up a SharePoint Server named "SPOINT1," then need to restore that server, the restored server had better be named "SPOINT1" or SharePoint might not work properly. With a Backup 2.0 solution, we're restoring the entire OS during disaster recovery—so things like the server name will remain intact. However, because we're simply writing disk blocks back to a disk, we're not limited in writing them back to the original physical server—which may have failed. We can write disk blocks to a virtual machine, allowing us to bring SharePoint back online even in an off-site recovery scenario.

Easier Management

Managing backup tapes is much easier because in some instances you might not even have backup tapes. Personally, I love the additional feeling of security I get from having copies of my backups off-site, and tapes are a great medium for it. Thus, your Backup 2.0 solution should definitely be able to move copies of its disk block repository to tape. But "shuffling" tapes is a lot less critical because from a tape perspective, you're only backing up *one thing:* the backup repository. A backup of the backup, if you will; thus, the tapes are literally just "Plan B" in the event your backup server dies or becomes inaccessible due to a data center problem or other facility disaster.

SharePoint-Specific Concerns

SharePoint, as I've pointed out previously, is not without its little quirks, and any backup solution used with SharePoint has to embrace those quirks and cater to them. Backup 2.0 is no different, so let's examine some of the major SharePoint-specific concerns and see how Backup 2.0 might deal with them.

Single-Object Recovery

Although SharePoint's versioning and Recycle Bin features can provide end users with their own single-item recovery, those features only go so far. You'll definitely want the ability to retrieve single items from your backups, and a Backup 2.0 solution—using a time-stamped disk block repository, as I've described—should be able to provide that capability. By mounting a specific point in time as a readable file system, you can attach all kinds of tools to browse the SharePoint repository and retrieve individual documents; in a worst-case scenario, you could restore the entire server to a physical or virtual machine and use SharePoint itself to access the files you need.

Some Backup 2.0-style solutions may even combine all the functionality you'll need: mounting the point-in-time snapshot, attaching to the SharePoint data stores, and presenting the SharePoint data in a graphical user interface (GUI) where you can browse the repository and select the files you want to retrieve.

Protecting Dependencies

SharePoint's configuration and data needs to stay properly synchronized; you can't restore a configuration database from 2 months ago *and* restore yesterday's content database and expect everything to work. That's why Microsoft's official guidance recommends against actually backing up the configuration and instead suggests that you "document" the configuration and re-create it manually in the event of a disaster. Because "manually" is why we have computers in the first place, right?

Backup 2.0 avoids that complexity by simply treating the *entire server* as a single unit of management, and by allowing you to restore *the entire server* to a specific point in time. That means you can always restore the content database *and* the configuration database *as it existed at that exact same point in time.* It's just in the nature of how Backup 2.0 works: You get *everything*, all the time.

Coming Up Next…

In the next chapter, I'll tackle a subject that's near and dear to my heart: virtualization backups. Virtualization has changed the way we think about our data centers, and has enabled a number of new, flexible computing scenarios that are saving businesses money and helping them to be more agile. At the same time, however, virtualization has upset many of the tried-and-true IT operations practices that have been developed over the years. In many ways, virtualization is "Computing 2.0," and Backup 1.0 just isn't a good fit. I'll explain how people have been *trying* to make Backup 1.0 get along with virtualized infrastructures, and suggest some ways in which Backup 2.0 could do a much better job and help fully realize the promise of virtualization.

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.