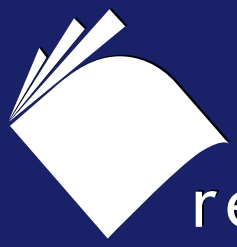

realtimepublishers.com™

The Definitive Guide™ To

Exchange Disaster Recovery & Availability

Paul Robichaux

Jim McBee, technical editor

Copyright Statement

Chapter 6: Implementing Disaster Recovery with Exchange

At this point, you should have a good theoretical understanding of disaster recovery technologies and practices. It's time to turn that theoretical knowledge into *practical* knowledge. To do so, this chapter focuses on how to implement the design changes and procedures that earlier chapters have discussed. This chapter relates to disaster recovery, and the next chapter will cover high-availability solutions.

Microsoft delivers a wealth of disaster recovery material on its Web site. Rather than duplicate that information, let's focus on a few key areas that aren't well covered or are outside the scope of Microsoft's guidance. Before we begin, take the time to look over

- The *Exchange Server 2003 Disaster Recovery Planning Guide* (http://www.microsoft.com/downloads/details.aspx?FamilyID=784BBEA2-28DD-409A-8368-F9914E993B28&displaylang=en), which describes planning and design issues involved in disaster recovery implementations for Exchange. This document is a little outdated, but might still be valuable as you work through your disaster recovery planning process.
- The *Exchange Server 2003 Disaster Recovery Operations Guide* (http://www.microsoft.com/downloads/details.aspx?FamilyID=A58F49C5-1190-4FBF-AEDE-007A8F366B0E&displaylang=en), which details specific procedures such as how to restore a database or storage group.

There are several disaster recovery scenarios. These can be expressed in a rough hierarchy according to the amount of data you need to restore to execute them; I prefer to group them according to the number of affected users. First, let's talk about the scenarios that affect individual users:

- Restoring one or more accidentally deleted messages. If you have deleted item retention turned on, and the items were deleted within the retention period, users can recover the messages themselves. If not, you can use a recovery storage group to mount a backup of the database and extract the needed messages.
- Restoring one or more accidentally deleted mailboxes. Exchange Server 2003 allows you to use the Mailbox Recovery Center to recover a mailbox that was deleted but has not yet been purged, provided you have deleted mailbox retention enabled.
- Restoring an accidentally deleted public folder. This scenario can be tricky; if you've deleted a replica from one server but other replicas exist, it's easy to add the replica back to the affected server. However, if you have only one replica, it's essentially the same thing as deleting a mailbox except that Microsoft doesn't include any tools to restore the public folder's contents.

The next step up the disaster recovery ladder is scenarios that affect groups of users on a given server. These include restoration of one or more databases or storage groups on a server, up to and including rebuilding and restoring a failed server by reinstalling or recovering Windows, reinstalling Exchange, and reloading mail data.

The third class of recovery solutions involve services; the important thing in this category isn't necessarily an individual server or data object; instead, the focus is on restoring the operation of some capability, such as Web access to email or the ability to perform AD queries. There are two basic ways to accomplish this class of recovery: you can either restore a failed service or you can create a new instance of it. We'll explore both of these methods.

Backing Up

There is a plethora of Exchange-aware backup products on the market, and each has its own strengths and weaknesses. The major players, such as Symantec and Computer Associates, have extensive documentation on how to make their products work with Exchange. Therefore, this section will focus on the basic mechanics of using ntbackup, the standard Windows backup utility, to back up and restore Exchange data. The reason for the focus on ntbackup is that everyone has it, and once you understand how to back up data with ntbackup, you can easily transfer your understanding to a different backup program if ntbackup doesn't meet your needs.

Microsoft, which can certainly afford any backup software and system, uses ntbackup on its Exchange servers. In addition, the company uses ntbackup to do disk-to-disk backup of its mailbox stores, then uses SAN transport to move the backup copies elsewhere on the SAN. (Microsoft has an excellent white paper called "Backup Process Used with Clustered Exchange Server 2003 Servers at Microsoft" that details how the company performs backup and restores of its Exchange systems.

System State

Before starting into the backup interface, let's take a second to explore the concept of system state, which is Microsoft's term for backing up the operating system (OS) and its data. For example, on an AD domain controller the system state includes the actual .dit files that contain AD's data; on all machines, system state data includes the OS's binary files, the contents of the registry, and other ancillary data (such as the contents of the certificate store) that is necessary to fully restore a computer. For example, when I perform a full backup of the system state on my primary Exchange server, it takes about 4.5 minutes and occupies about 600MB on disk. Because it's a relatively small data set and it's critical to keep around for restores, you should be certain to make regular system state backups.

However, Microsoft made a change in WS2K3; system state backups are captured using the Volume Shadow Copy Service (VSS). When a VSS-aware backup program (such as the WS2K3 version of ntbackup) requests a VSS copy of data, the VSS component will temporarily freeze I/O requests to the volume that contains the data being requested. If you use VSS to capture a system state backup and your Windows installation is on the same volume as Exchange, VSS will freeze I/O to the volume when the backup starts, but it won't unfreeze it until the entire backup finishes. This process isn't necessarily harmful, but it can cause problems if the backup doesn't complete normally. Accordingly, unless you're using VSS to back up your Exchange data too, schedule system state and Exchange backups separately.

Let's walk through the basic process for backing up your Exchange server with ntbackup. First, you need to install the Exchange-aware version of ntbackup. This installation happens automatically when you install the Exchange System Manager (ESM) on a computer (or when you install Exchange itself, for that matter). Ntbackup allows you to back up an Exchange server remotely as long as you're running the Exchange-aware version of ESM (you can't back up system state data remotely, though). Once the installation's done, making a backup is quite simple:

- Launch ntbackup.
- If you see the Backup Wizard, cancel it.
- Select the Backup tab. You'll see a view similar to the one that Figure 6.1 shows.
- Expand the Microsoft Exchange Server item in the left column. You'll notice that it contains one entry for each of your Exchange servers.
- Expand the first server that you want to back up, then select the storage groups and databases that you want to back up.
- Use the *Backup destination* and *Backup media or file name* fields to specify where you want the backup to go.
- Click Start Backup.

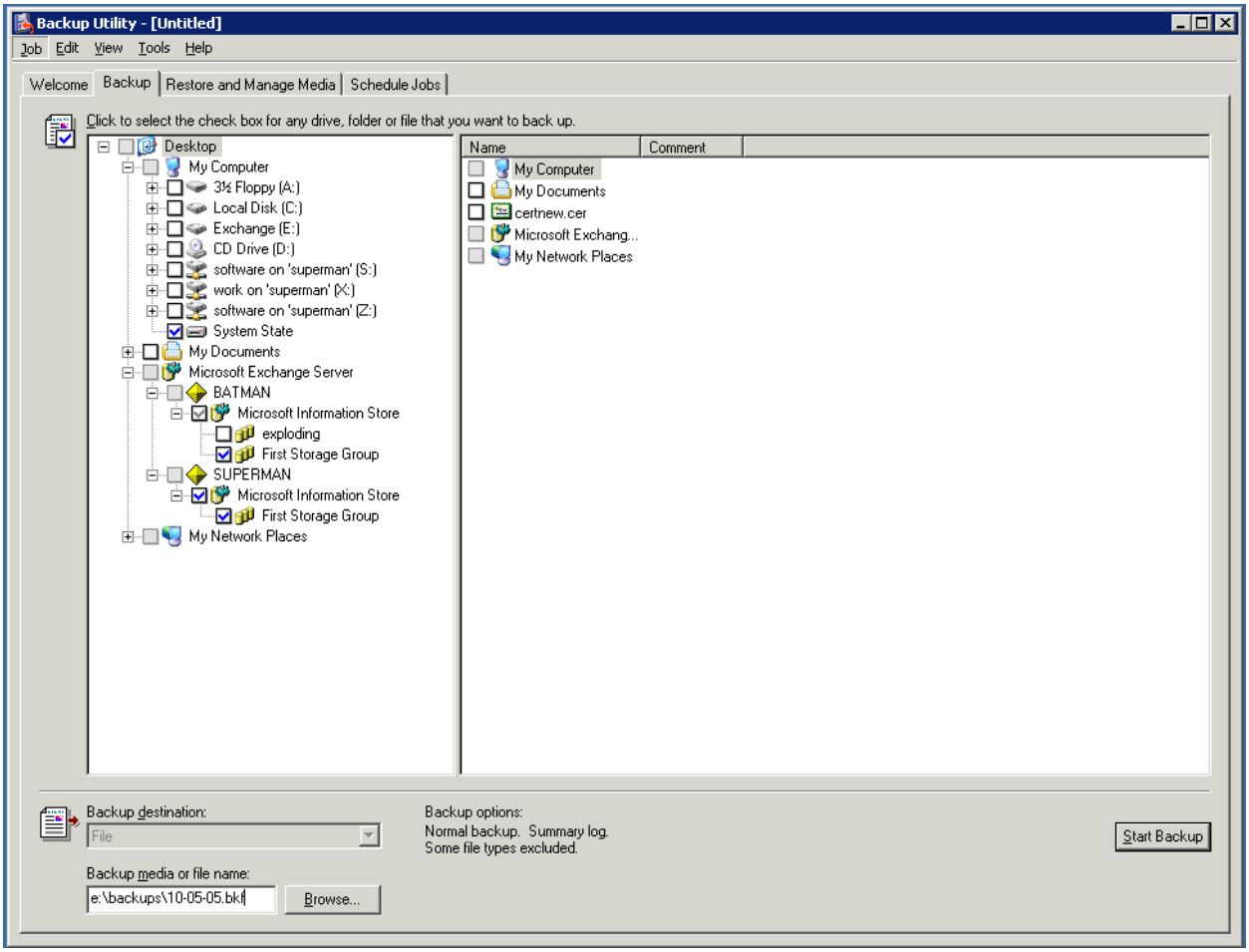

Figure 6.1: The primary interface for ntbackup.

As you can see, this process is simple and ntbackup is a basic and easy-to-use tool. As a refreshes, let's review how the online Exchange backup process works:

- The backup program calls the Exchange backup APIs (which are more properly referred to as the ESE APIs) to indicate what type of backup is requested (full, incremental, and so on).
- The backup program asks for a list of databases that can be backed up; it may or may not show this list to the end user, depending on whether it's an interactive or scheduled job.
- For each database to be backed up, the backup program calls the ESE API to open the data file (which prevents Exchange from writing new data to it), then repeatedly calls a routine to read a block of data from the database. This read cycle has some additional logic in it:
    - ESE reads each page from disk.
    - It computes a checksum for the page's data and compares it with the checksum included in the on-disk copy of the page. If the checksums match, great; if not, the backup fails with a -1018 error.
    - It checks the pointers on the page to ensure that they're all pointing to valid pages. If any pointers are dangling, or pointing to incorrect pages, the backup stops.
- After all the databases are backed up, the backup program releases the backup files and requests access to the transaction logs. During the preceding steps, new transactions can still be accepted, they are simply logged using the existing logging mechanism. During this step, new transactions are applied directly to the database. Each individual log file is opened, backed up, and closed separately.
- When all the log files have been backed up, if a full or differential backup was requested, the log files are truncated—that is, the old log files are removed from disk.

Ntbackup is explicitly designed to be very simple, but there are a few tricks that you should keep in mind. First, you can use a special unbuffered mode with the command-line version of ntbackup to increase your backup throughput—the /fu switch (which stands for file unbuffered) forces ntbackup to use unbuffered I/O. This setup is primarily beneficial when you're backing up to disk, but even with tape systems, it may significantly increase your backup throughput.

That naturally leads to the next question: how can you use ntbackup from the command line? A quick check of the online ntbackup documentation at http://www.microsoft.com/resources/documentation/WindowsServ/2003/standard/proddocs/en-us/ntbackup_command.asp shows many command-line switches, but nothing that looks like a way to back up Exchange data. The answer lies in the fact that you can create a file of instructions for ntbackup and then run *that* from the command line. This process turns out to be quite simple once you know the (undocumented) trick, provided that you want to do disk-to-disk backups (the interface for choosing a tape device or media pool requires you to specify their GUIDs).

The first, and simplest way, is to use the interface that Figure 6.1 shows to specify what you want to back up. Once you've done so, use the Job menu's Save Selections As command to save a backup selection (.bks) file. This file is actually a plain text file in which each line specifies the path to the store you're backing up. Here's an example:

```
JET BATMAN\Microsoft Information Store\First Storage Group\Sales
JET BATMAN\Microsoft Information Store\First Storage Group\Legal
JET SUPERMAN\Microsoft Information Store\First Storage
Group\Public Folder Store (SUPERMAN)
```

This command tells ntbackup to back up two mailbox databases on BATMAN and a public folder database on SUPERMAN. The JET keyword is required; this keyword tells ntbackup to use the Exchange API to perform the actual backup. Apart from that requirement, the format is self-evident, so you can easily create your own .bks file.

Once you have a .bks file, you can tell ntbackup to use it with the @ command-line switch. For example, the following command tells ntbackup to use the backup selection file in e:\exchsrvr\mdbdata\Monday-full.bks and to store the data in a .bkf file named Monday-gen1-full.bkf:

```
Ntbackup backup systemstate "@e:\exchsrvr\mdbdata\Monday-
full.bks" /fu /f "x:\backups\Monday-gen1-full.bkf"
```

As a bonus, the systemstate keyword does what you'd expect.

When you include the systemstate keyword, you're going to back up the system state of the machine on which you're running ntbackup. If you use a .bks file to back up data from other Exchange servers across the network, it is important that you're aware of this consideration.

Keep in mind that you can also use ntbackup's scheduler (Figure 6.2) or the system's task scheduler to schedule backup jobs. The combination of .bks files and the task scheduler is adequate for many single-server backup tasks, although larger numbers of servers and more complex environments may benefit from having a more powerful backup program.

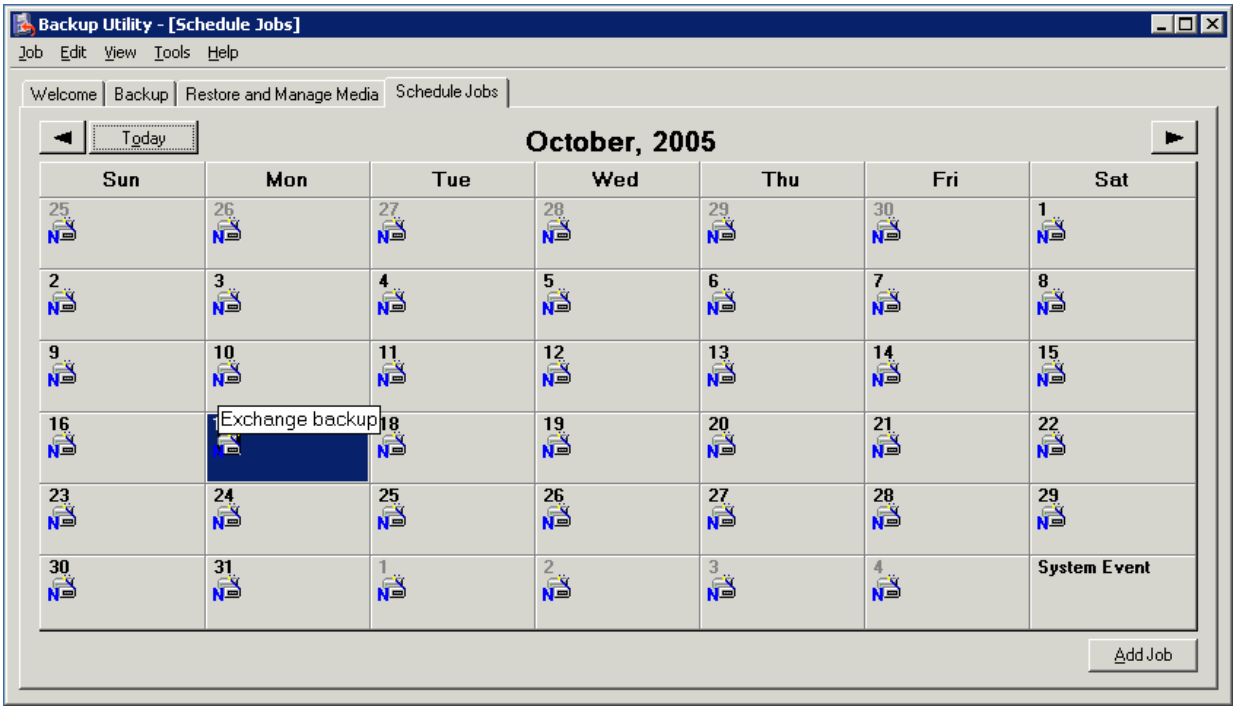

Figure 6.2: The ntbackup scheduler showing a daily backup job.

So what is backup throughput like with ntbackup? The answer is a resounding "It depends." With disk-to-disk backups on an otherwise idle server, you can easily get throughputs of 1GB per minute or more. You might have to make a few registry tweaks to reach this speed, however. The excellent "Backup Process Used with Clustered Exchange Server 2003 Servers at Microsoft" white paper describes the changes you need to make. Basically, you can add registry values beneath the HKEY_CURRENT_USER\SOFTWARE\Microsoft\Ntbackup\Backup Engine key to control how many buffers ntbackup uses (and how much space is allocated for each). You should perform a thorough test of these values in your environment to determine the optimum configuration for the amount of data you're backing up and the speed of your backup subsystem.

Avoiding Restores

Conventional disaster recovery is a goalkeeper sort of technology—something that you use only as a last resort. Along those lines, one of the goals of this chapter is to emphasize that you can *avoid* restores in many situations in which you might think they are necessary.

What "Deleted" Really Means

Most non-administrators think that when you delete a message, it's gone forever. Such isn't always the case, as most email administrators can attest. Understanding how Exchange treats deleted items is an important part of any strategy to reduce the number of restores. There are three stages of a deleted message to consider.

The first stage occurs when the user removes an item from his or her client. Outlook and OWA will move the deleted message to the Deleted Items folder (as will some IMAP and DAV clients). The item still exists, and it's still accessible to the user because it's in the Deleted Items folder.

The second stage occurs when the item is removed from the Deleted Items folder. This removal may occur because the user's emptied the folder or the user held down the Shift key when deleting the item (which bypasses the Deleted Items container altogether; this method is known as a *hard delete*). Items in this stage are still in the mailbox database; they're actually in the deleted item retention container, more commonly known as the dumpster. Each mailbox has such a container; items in this container have the MAPI PR_DELETED_ON property, which is how the store tracks them. Every day, as part of the scheduled maintenance period, the store removes dumpster items that have hit the end of the deleted item retention period. Figure 6.3 shows the event ID logged when online maintenance completes. At that point, the only way to get the item(s) back is to recover them from an older backup of the database.

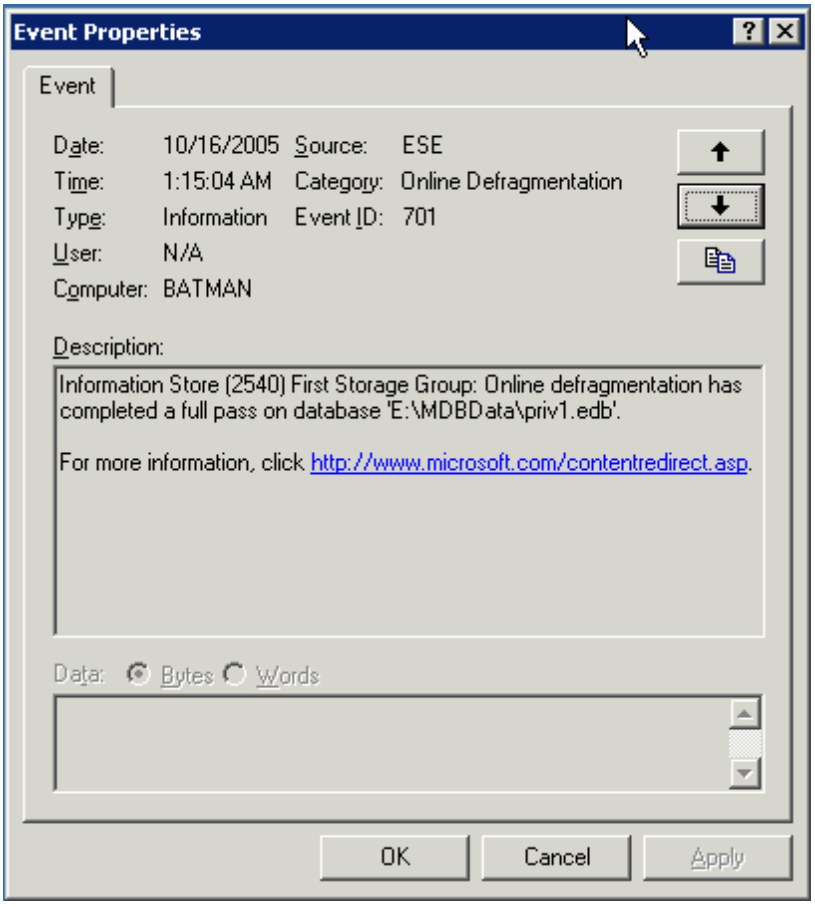

Figure 6.3: The store logs event ID 701 when online defragmentation completes.

Interestingly, the item actually stays in the database until the deleted item retention period expires; at that point, the pages used to store it are marked as free but the item's data isn't deleted from those pages. Free pages can be reclaimed by the store at any time; you can also tell most backup programs (including ntbackup) to zero free pages during backups so that deleted message contents can't be recovered.

Recovering Messages

If you're interested in recovering deleted messages—or, more precisely, letting users recover their *own* deleted messages—you need to look into the deleted item retention period on your mailbox databases. You can view and set these properties on the Limits tab of the properties dialog box for any individual mailbox database (see Figure 6.4).The *Keep deleted items for (days)* and *Keep deleted mailboxes for (days)* are self-explanatory—they govern the retention period for those object types. The *Do not permanently delete mailboxes and items until the store has been backed up* flag works a little differently. By default, deleted items will be removed from the store during the next online maintenance period that occurs after the end of the retention period. If you select this check box, those items won't be removed until the next maintenance period that follows a successful backup of that database.

> You can also set these limits with a system policy that applies to any or all of your mailbox databases; this option is a better idea than applying settings to individual computers because you want to have a consistent retention policy.

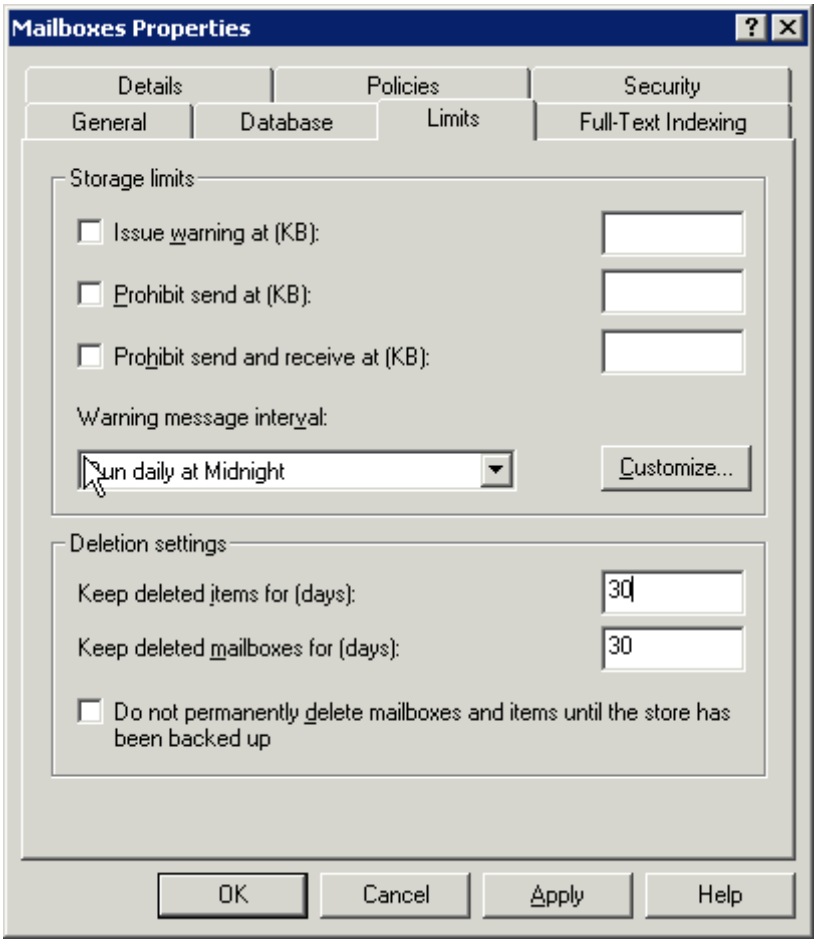

Figure 6.4: Setting deleted item retention settings.

Once you've applied the appropriate retention settings, you're done on the server side. By default, Exchange Server 2003 is set to 7 days; most sites that want to utilize deleted item retention want to keep deleted items for longer; 14- to 30-day periods are typical.

Keeping deleted items consumes disk space; the longer you keep items, the more space you'll need. Because this space isn't counted against users' mailbox limits, you should ensure that you have enough free space to handle 15 to 25 percent more than the amount of mail data you expect to have—just in case.

From the user side, recovering items is simple. In Outlook, you can use the Tools, Recover Deleted Items command. This command will work for items that have been deleted from the Inbox. You can optionally allow users to recover hard-deleted items or items deleted from other folders by adding a new DWORD value named DumpsterAlwaysOn to the HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Exchange\Client\Options key on the client. You can also recover deleted items from within OWA. To do so, open the OWA options page, and scroll to the bottom to click View Items in the Recover Deleted Items section.

Recovering Mailboxes

Exchange Server 2003 implements deleted item retention for mailboxes, too, not just individual mail items. Deleting a mailbox doesn't remove it from the store; instead, the connection between the user account and the mailbox object in the database is severed. You can easily reconnect such mailboxes; they appear with a red X icon in the Mailboxes view of each mailbox database. (If you don't see the mailbox you're looking for, you might need to right-click the Mailboxes view and use the Run Cleanup Agent command to force ESM to update its list of disconnected mailboxes.) Simply right-click the deleted mailbox, and select Reconnect. Remember that you can only reconnect a deleted mailbox to an account that doesn't have a mailbox connected to it—an AD account can be linked to only one mailbox at a time.

If you have many mailboxes that were accidentally deleted, you can use the Mailbox Recovery Center to bulk-reconnect them. To do so, select the Mailbox Recovery Center item in ESM (it's in the Tools node), then right-click this item and use the Add Mailbox Database command to select the stores that contain the mailboxes you want to recover. The right ESM pane will update to show you the list of available (that is, disconnected) mailboxes, and you can select the ones you want to reconnect. Right-clicking them gives you the option to reconnect them.

Recovering a Deleted Public Folder

Recovering the contents of a deleted public folder can either be easy or difficult. The difference? Whether you have another replica of the public folder stashed away somewhere. Adding replicas of folders is very straightforward:

- Launch ESM.
- Expand any administrative group that contains a replica of the folder you want.
- Expand the Folders and Public Folders node.
- Right-click the folder for which you want to add a replica, then choose the Properties command.
- Select the Replication tab.
- On the Replication tab, click Add, then select the public folder database that you want to contain the replica.

Naturally, these steps won't do you any good unless you take them *before* your folder's deleted, and they won't help if you only have a single server. If you've lost a folder with critical data and there are no replicas, you'll have to recover the data. The simplest way to do so is with a third-party tool such as Quest's Recovery Manager for Exchange or Ontrack's PowerControls. Both of these tools allow you to restore data from a dismounted mailbox or public folder database. Failing that, you can restore the database, but you can't use a recovery storage group, so you'll have to plan to take the existing database offline while you do the recovery—probably not a popular move.

Of course, some kinds of public folders have data that isn't worth backing up. An example is the system's Schedule+ Free/Busy folder, because clients republish that data if it's missing. Another example: NNTP newsgroup folders—their contents can easily be backfilled from your NNTP server, provided that the articles you need are still available on the server.

Restoring a Single Database or Storage Group

Exchange restoration actually involves two phases. First, you have to retrieve the database and log files that you want to recover and return them to the correct location. Once that's done, you may have to replay log files to return the database to a consistent state; in some cases, you may have to tinker with the log files to affect recovery.

Restoring with Ntbackup

Restoring data with ntbackup is done using its built-in catalog of known backup sets. When you back up data on a server, ntbackup updates the on-disk catalog to reflect what was backed up and where it was stored. If you're backing up to tape, there is also an on-media catalog that can be used to reconstitute to the on-disk catalog. If you're restoring data onto a server other than the server on which the back up was run, you'll normally have to start by using the *Catalog a backup file* command in the Tools menu to update the catalog to reflect what is in the backup files. Once the desired backup set appears in the Restore and Manage Media tab (see Figure 6.5), you can click Start Restore to begin the restoration process.

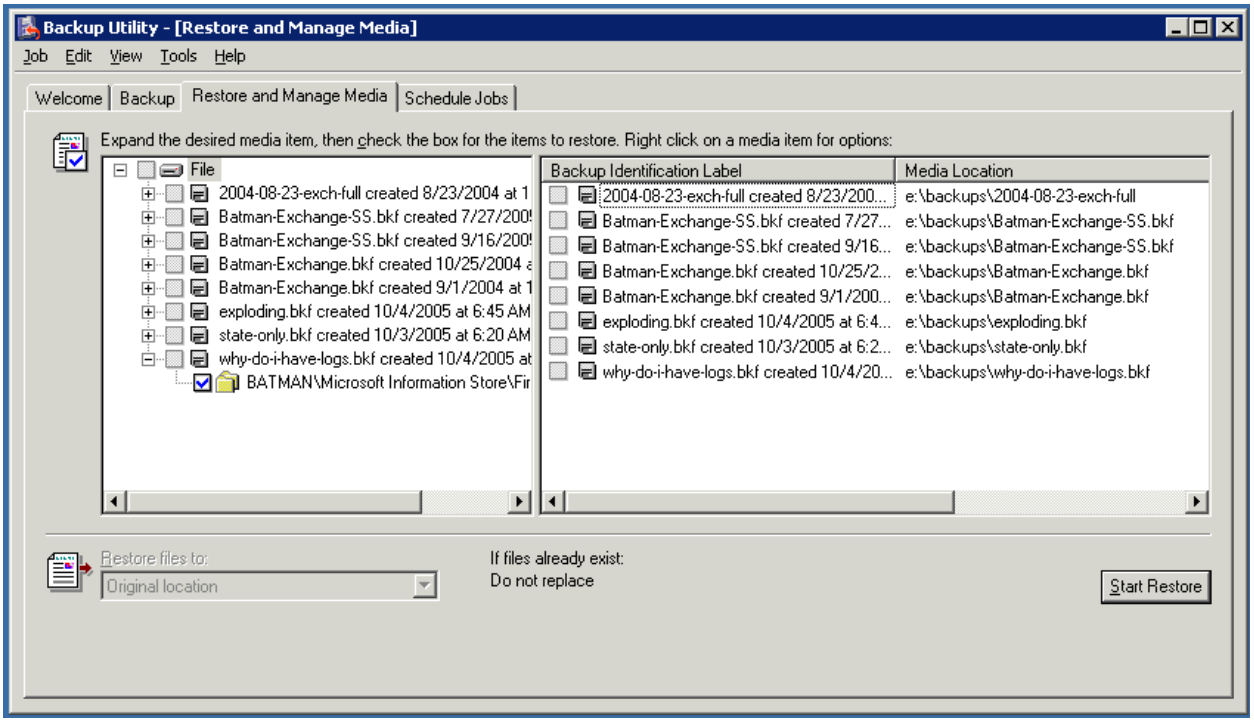

Figure 6.5: Selecting the backup set to restore.

Clicking Start Restore displays the Restoring Database Store dialog box (Figure 6.6), which is where you need to make three important choices.

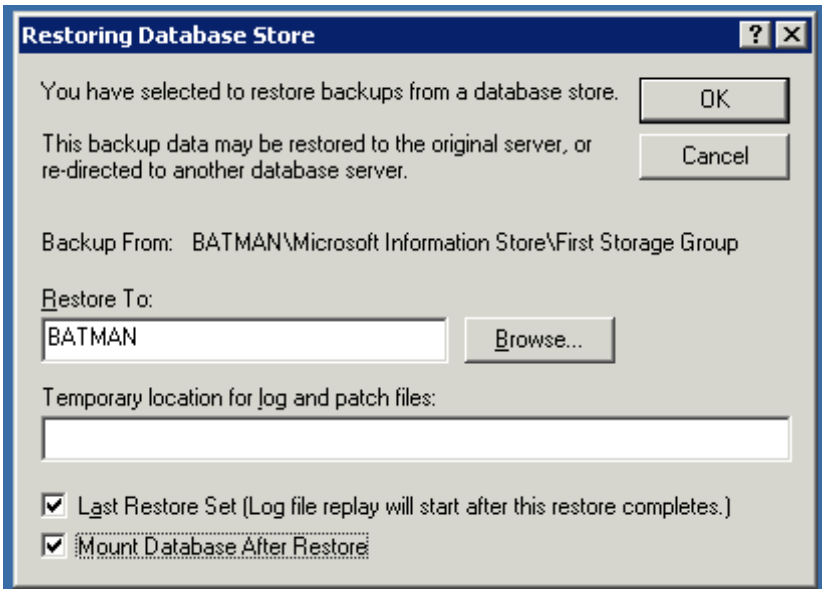

Figure 6.6: Restoring a database.

The first choice is where you're going to restore the data. By default, ntbackup will offer to restore to the server on which you're performing the restore, but you can redirect the restore to another computer by selecting the computer using the Browse button or entering its name. However, you must remember that if your server has a recovery storage group (RSG), ntbackup (and all other backup programs that use the ESE backup APIs) will automatically redirect the restored data to the RSG. If you don't want this action to take place, see the section below on RSGs to find out how to override it.

The second choice is where to put the log files during the restore. You don't want to put them in the default location for the storage group you're restoring, as that may accidentally overwrite existing logs. Use the *Temporary location for log and patch files* field to specify a full path (no UNC paths allowed) to a local disk on the server on which you're doing the restore. C:\temp is often a safe bet because you shouldn't be keeping anything critical there.

The third choice involves what you want done after the database files in this backup set have been restored. The *Last Restore Set (Log file replay will start after this restore completes.)* check box controls whether log playback will begin immediately after the restore. If you're restoring a backup with only one component (such as a full backup of an entire storage group), selecting this box allows the store to immediately begin playing back recovered log files when the restore finishes. If you leave the box clear, as it is by default, the logs won't automatically be played back. In general, you should select this check box when you've restored the final data set you need. For full backups, this is pretty much self-evident; for incremental or differential backups, don't check the box until you're restoring the last incremental set that you intend to restore. For example, suppose that you do a weekly full backup on Monday, followed by daily incrementals. If you're restoring Monday afternoon, you would select the check box so that log playback can begin right after the restore. If you were restoring Thursday night, you would restore the Monday full backup—with the box clear—then the incremental sets for Tuesday, Wednesday, and Thursday.

There's an additional check box—*Mount Database After Restore*. It's pretty clear what it does—so why wouldn't you select this item? The primary reason is that as soon as you mount the database, Exchange will start trying to play back any log files that haven't previously been committed. If you've got all the log files, that is not a bad idea, but if you still need to restore some of them, or if you have to run diagnostic tests or repairs on the database using *eseutil* or *isinteg*, you won't want the database mounted before you do anything with it.

There is no practical difference between restoring a single database and all the databases in the storage group—the only real difference is that restoring a storage group explicitly recovers the transaction logs unless you clear the *Log Files* item in the right pane of ntbackup (see Figure 6.7). Remember that when you restore multiple databases in the same storage group, you shouldn't select the *Last Restore Set (Log file replay will start after this restore completes.)* check box until you've restored the last database in the set.

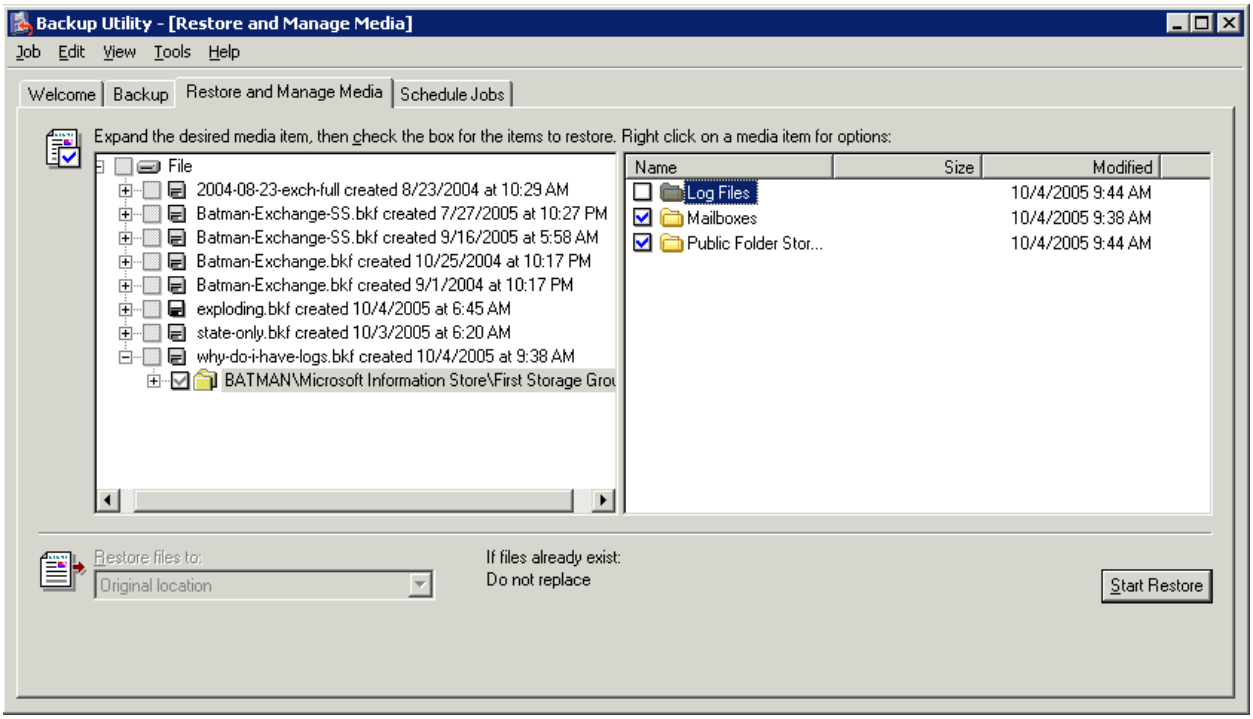

Figure 6.7: Selecting databases from a storage group without the log files.

To play back log files manually, use the *eseutil* tool. The /cc switch tells eseutil to play back log files for a restored database to attempt to make the database consistent. This process, known as *hard recovery*, normally happens when the Last Restore Set check box is selected; as mentioned earlier, you might not want to let your backup software do this automatically.

Restoring an Entire Server

There are times when you have to restore an entire server. One of the big advantages of third-party backup products is that they generally offer automated tools for copying all of a server's contents into an easy-to-use restore package. This feature goes by various names, including automated system recovery and intelligent disaster recovery. Fundamentally, what these options do is gather the system state data, whatever server-specific data you select (such as your Exchange databases), and any drivers needed to operate your backup subsystem. The backup program then generally creates a bootable DVD-R or CD-R with the drivers and backup software; to execute a recovery, you boot using the recovery tool and restore your data.

Rather than go through the specifics of using a particular backup product to do this, the following list highlights considerations for planning whole-server restorations:

- Remember to factor in the amount of time that it will take for you to restore Windows on the server, particularly if the server is a domain controller. You'll normally accomplish this with a system state backup.
- If you're restoring a server that was an AD domain controller or Global Catalog server, you can restore whatever version of the local data you have, then allow other domain controllers to update it. However, if you have only one domain controller, and Exchange is installed on it, you'll need to restore AD and ensure that it works properly before restoring Exchange. This process can obviously add a good bit of time to your restore operations.
- If you're restoring the server on your own, you'll need to reinstall Exchange using the /disasterrecovery switch, which tells Exchange's setup program to install the Exchange binaries but not to create new AD objects for the server. Of course, you'll need to be prepared to install any Windows or Exchange service packs that you had on the computer before the failure. If you're using an automatic disaster recovery method, you may not need to separately reinstall Exchange.
- Don't forget ancillary services that you may have running on your Exchange servers. Examples include antivirus and anti-spam software, the Windows Certificate Services module, and other tools or data sets that you may need on your server. It's a good idea to tailor your recovery plan to allow for operations without these tools; that way you can bring up the server as quickly as possible and restore the additional services once basic services have been restored.

Using Recovery Storage Groups

In Exchange 5.5 and Exchange 2000 Server, databases are designed to be used only on the machine on which they were created. There are several reasons for this; primary among them is that the database contains embedded information about the machine name, the database's location on disk, and the machine's path in the directory. However, this setup is ineffective from a disaster recovery point of view—in many circumstances, the only way to restore is to use a machine with the same name and disk configuration as the original.

Exchange Server 2003 relaxes this requirement quite a bit by introducing the concept of the RSG, a special-purpose storage group that, once created, can mount a database from any Exchange 2000 or Exchange Server 2003 server in the same administrative group. This addition greatly simplifies restoration, provided that you have a complete backup of the database. When a database is mounted in the RSG, you can access the data with ExMerge, but users can't directly log on to it, and new mail can't be delivered to it. One common scenario during a failure is to create a new, empty database for a group of users so that they can originate and receive mail, then (when time and circumstances allow) recover the real database to an RSG and move previously delivered mail to the new database.

The following list highlights the key restrictions to remember when working with RSGs:

- RSGs require that your AD setup be intact.
- You can't connect mailboxes in an RSG to individual user accounts, so they're not useful for recovering deleted mailboxes.
- You can only recover databases within the same administrative group. As a practical matter, this isn't much of a restriction because the majority of Exchange sites are only using a single administrative group anyway.
- You can only recover databases from one original storage group at a time but you can mount multiple databases from that storage group. Let's say that you're recovering a server that had two storage groups (SG1 and SG2), each with two databases (SG1DB1, SG1DB2, SG2DB1, and SG2DB2). You can recover SG1DB1 and SG1DB2 at the same time, but you can't recover SG1DB1 and SG2DB1 at the same time.
- When you mount a database in the RSG, if it's from an older version of Exchange, the database may be physically upgraded. This occurrence is a problem if you recover an Exchange 2000 database to an Exchange 2003 RSG; once the database has been mounted in the RSG, you can't copy the same database back to the original server without upgrading the server.
- You can't restore public folder data with RSGs. This limitation isn't a big deal because you can often recover a single server's public folders by backfilling them from another replica (provided you have another replica).

Creating an RSG

The basic process for creating an RSG is simple: launch ESM, and right-click the server on which you want to put the RSG. From the resulting context menu, choose the New, Recovery Storage Group command. When the RSG properties dialog box appears (see Figure 6.8), name the RSG; you can also change the paths for the RSG's transaction log files.

Bear in mind that the name matters! When you restore a database, if its original storage group exists on the server, the RSG has to have a *different* name. If the original storage doesn't exist on the recovery server, the RSG has to have the same name as the original storage group. To clarify, suppose you want to restore a database named Execs01 in a storage group originally named DETROIT-SG01, on a server named DTWMBX01. If you Restore Execs01 back to DTWMBX01, the RSG has to be named something else (such as Recovery). If you restore Execs01 to a different server, the RSG must be named DETROIT-SG01.

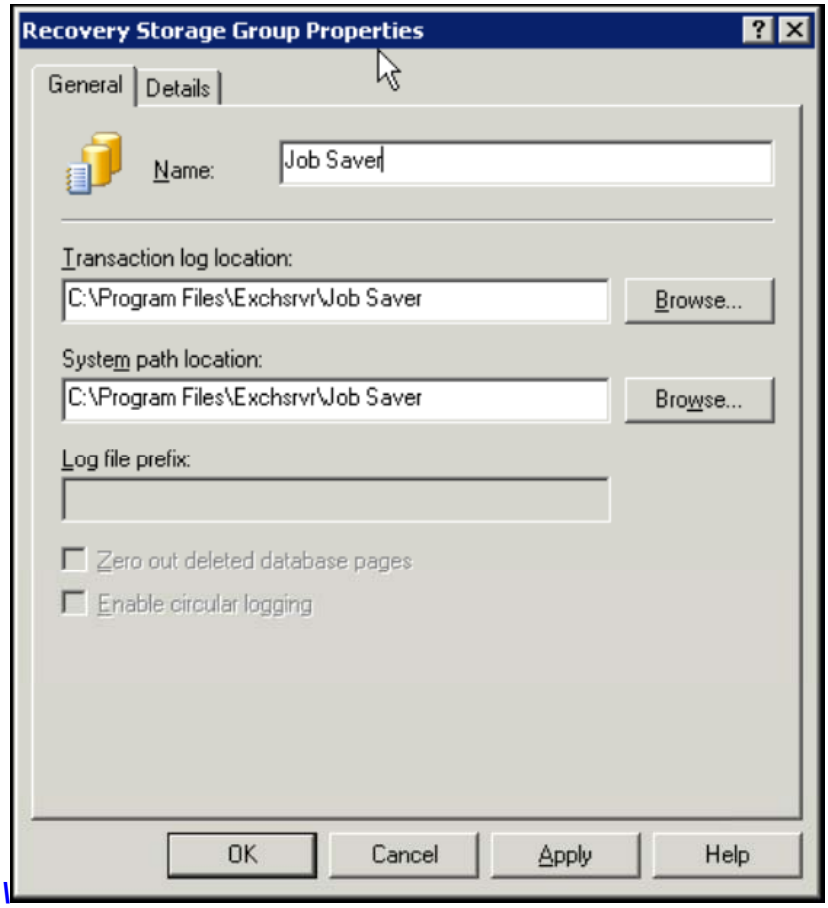

Figure 6.8: The RSG properties dialog box.

You shouldn't create an RSG on your server until you need it because having an RSG present affects the way Exchange restores work. In a normal restore, the Exchange backup API will restore the database files from a backup set to their correct location, then initiate log playback when you restore the last set of files required for the storage group or database that you're restoring. When there is an active RSG, if you restore a database that is present in the RSG, the database files and logs are automatically moved to the RSG's path. If you try to restore a database that's *not* already in the RSG, the restore operation stops. This process prevents you from accidentally overwriting your databases during a restore, but it also means that if you leave an RSG on your server, you can't restore any databases until you add them to the RSG. In practice, this may not be a severe limitation.

Adding Databases to the RSG

Before you can actually complete a restore, you must add the databases you want to restore to the RSG. To do so, right-click the RSG and use the Add Database to Recover command; doing so displays the database selection dialog box that Figure 6.9 shows. ESM intelligently filters the items you see here; it hides any databases that were created with a newer version of Exchange than the one you have on the RSG server, and if you've already mounted a database in the RSG, only other databases from the same original storage group are shown.

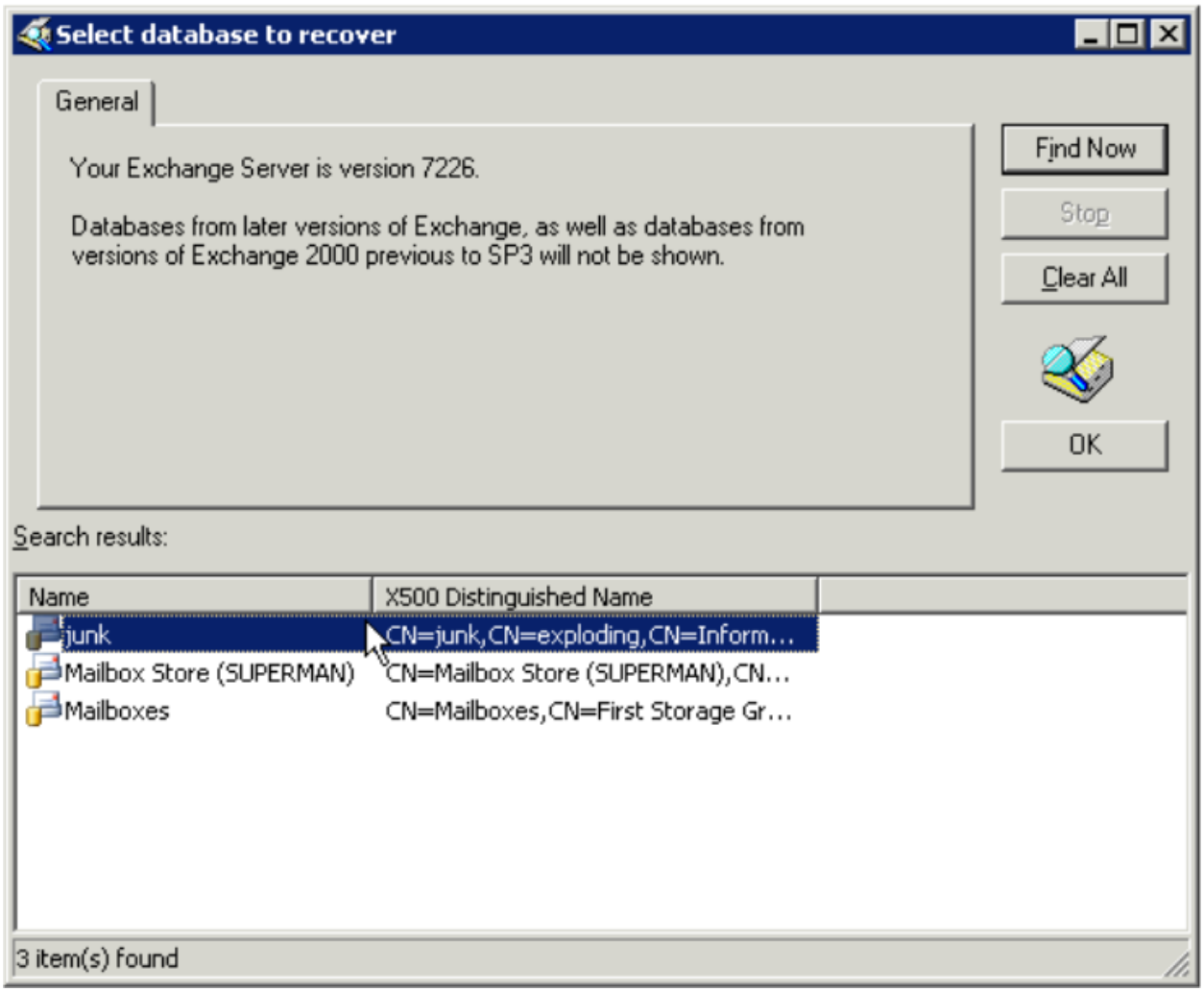

Figure 6.9: Selecting a database to add to the RSG.

To actually prepare a database for restoration, you simply need to select the database, and click OK. The system then displays the database properties dialog box in which you can change the name of the displayed database and the paths to where the EDB and STM files are stored. If you're going to copy the RSG's files to the original location, remember that they need to have the same name as the original files.

Recovering Data from an RSG

The obvious first step to using an RSG is to restore the database files themselves. You can do so with the steps in the previous sections; once you get the EDB and STM files back on the disk, you either need to move them to the path specified in the RSG properties or create the RSG at the correct location.

Once you've done so, you can use the Exchange 2003 version of ExMerge to extract data from the RSG. Remember that the RSG isn't directly accessible to clients, so the only way to get data from the RSG's databases to the client is to copy it to a PST file with ExMerge and then import it back into the mailboxes. If you're merging data from a recovered database back to an existing one, you may have to adjust permissions on the target mailbox database; normally, administrators are denied access to individual mailboxes by means of a deny entry on the access control list (ACL) for each mailbox database. ExMerge can read data from a database mounted in an RSG, but it can't copy that data to the target database unless you override the default deny entry. To do so, Microsoft recommends that you create a security group for recovery operations, then add the administrators whom you want to be able to recover data. Grant the security group Full Control permissions on the database into which you want to merge data.

Setting Up Dial-Tone Recovery

The process behind enabling dial-tone recovery is pretty straightforward. First, you remove the database files of the mailbox database (or databases) that you want to restore so that Exchange creates empty files. To do so, unmount all the databases in the storage group that contains the database you're repairing, then copy the transaction logs to a safe place and remove, or rename, the EDB and STM file databases you want to repair. Next, mount the databases again; ESM will warn you that it's going to create a new empty database. That's just what you want.

The effect of this first step is simple: when users connect to the database, their mailboxes are created anew (losing all rules, forms, and views into the bargain). From that point forward, users can send and receive mail just as they would have normally. The big difference, of course, is that they no longer have any of their previously sent or received mail. Thus, the reason it is called "dial-tone" messaging: it's a basic service, not a complete restoration of the same service level present before the failure.

The second step is also clear: while users are working with their new, empty mailboxes, you can restore the previous databases to the RSG. (Microsoft recommends putting the RSG on the same logical volume as the real databases to speed copying data between the two locations.) This process may be quick or it might take awhile—it all depends on your recovery processes and how well you execute them. Because the databases may require repair, this step may actually take longer than you expect. After you restore the database, you should mount it and then dismount it; doing so forces the store to play back any log files.

At this point, your users are still using the dial-tone database, and their older mail is in the repaired database. There are two ways to combine the data: you can copy from the dial-tone database to the original database, or vice versa. However, as the dial-tone database is almost certainly smaller, the merge operation will be much faster if you copy data *to* the restored database. However, this requires some tricky database swapping: users can't connect to the RSG database, so you have to move the RSG files to the correct location (thus restoring access to the original databases) while also moving the dial-tone databases to a safe location (so they're not overwritten).

After that's done, your users can log on to their old mailboxes, and you can run ExMerge again to move data from the dial-tone database into the corresponding mailboxes on the newly restored database. It's a good idea to merge only one mailbox at first, just to make sure that you've moved everything to the right location. If that merge goes OK, you can move on to doing more ambitious merges.

Using Outlook 2003 During Dial-Tone Recovery

If you're using Outlook 2003 in cached Exchange mode, your users will see an odd message when you initiate a dial-tone recovery: *Exchange is currently in recovery mode. You can either connect to your Exchange server using the network, work offline, or cancel this logon.* Deciphering this message is easy if you know the trick.

In cached Exchange mode, Outlook keeps an OST file. This is no different from previous versions of Outlook; the difference is that older versions of Outlook used a per-MAPI-profile encryption key for the OST file. Change the mailbox—as you'd have to during a dial-tone operation—and the old key becomes invalid, thus making it impossible to access data in the OST file until the original mailbox database is restored.

In Outlook 2003, things work a little differently; Outlook 2003 was expressly designed to accommodate dial-tone operations. When you connect in online mode, you get the contents of the mailbox on the server—and during a dial-tone recovery, that's all you get. The previous mailbox data is still safely in the OST file; you just can't access it while Outlook is connected to the server. If you need access to the OST data, you can start Outlook in offline mode, at which point the OST file will become available again.

What if you want access to both the dial-tone and old OST mailbox data at the same time? Easy. Start in offline mode; create a new PST file, then copy the OST file's contents to the PST. Quit and relaunch Outlook in online mode, then open the PST. You can even drag items from the PST file to the dial-tone mailbox, but doing so may result in duplicate items when you merge data back into the production database after the recovery.

Overriding RSG Recovery

When there is an RSG on your server, your backup programs will restore databases into the RSG. In most cases, this is what you want, because by using an RSG you get a nondestructive way to mount a database and copy data from it. However, you may occasionally want to override this behavior and restore data directly to its normal location. There are two ways to do so: you can remove the RSG (which means that when you *do* want to use it, you'll have to add all the databases back to it), or you can add a temporary registry key to override the RSG restore behavior. Microsoft strongly recommends using this key only in a test environment; if it's present, and you do a restore with the expectation that your data will go to the RSG, it may overwrite something important.

Having said that, if you still decide that you need this functionality, you can add a DWORD value named Recovery SG Override to HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\MSExchangeIS\Parameters System. Set it to 1 to enable the override behavior; remove it, or set it to 0, to keep the default RSG behavior. Be very careful if you decide to do this; it's easy to accidentally overwrite a production database with an older version! If the previous version of database you're restoring is mounted, then you'll get an error message because you can't overwrite a mounted database. If not, however, your backup software will happily replace the original database, and that's probably not what you had in mind.

Summary

Knowing how to plan and execute disaster recovery operations for Exchange is probably the most important single aspect of your overall messaging operations plan. Given the fact that you call on your disaster recovery operations only when something has gone bad, being able to flawlessly execute your plan is a very, very good idea.

Content Central

Download Additional eBooks!