

# Some Like It Hot

## Backing Up Novell eDirectory 8.7

by Linda Kennard and Brian Hawkins



**b**acking up your company's Novell eDirectory tree is essential to ensuring that your company's network can continue to perform under any circumstances. However, until Novell eDirectory 8.7, backing up the directory could be a tricky process. Happily, eDirectory 8.7 includes a new cross-platform backup and restore utility called the Backup eDirectory Management Tool (eMTool). The cool thing about Backup eMTool is that it's hot, or rather it enables hot backups.

Backup eMTool enables you to back up the entire eDirectory database on the server on which you run the tool. Regardless of how many changes to the directory occur while this backup is running, Backup eMTool captures a complete view of the eDirectory database that's consistent with the moment you start the backup.

The other cool thing about Backup eMTool is that it's fast. James Whitchurch, Novell director of software engineering for eDirectory, estimates that backing up an 8 GB eDirectory database for a tree with 12 million objects takes no more than two hours. Restoring such a tree takes roughly the same length of time, Whitchurch says. Whitchurch bases these estimates on the total length of time involved in a backup and restore process—including the time you spend configuring the backup and gathering the backup files that you need to restore the directory.

Novell engineer Brian Hawkins, who developed Backup eMTool, says that Novell has tested Backup eMTool on an eDirectory tree just this size. The actual backup—that is, the time it took to back up data to disk—took approximately only 30 minutes. Restoring the eDirectory tree took only about five minutes longer, Hawkins says.

### UP FRONT ABOUT BACKUPS

Regardless of the server platform where you choose to install eDirectory 8.7, its installation program automatically installs Backup eMTool as part of the eMBox tool set. (For more information about eMBox, see the "Tier Three" section in the "Novell iManager: Keeping eDirectory Management Simple" article on p. 7.) After you have installed Backup eMTool, you can access the tool using either Novell iManager or the eMBox command-line Java client. Novell iManager provides a GUI and enables you to access and run Backup eMTool from

outside your firewall. The Java client enables you to run batch files, which in turn allow you to run backups unattended.

To use Novell iManager to start a backup process, you choose Backup from the eDirectory Maintenance Tools list and then follow the prompts in the Backup Wizard. The prompts are pretty basic. For example, you are prompted to specify the server on which you want to perform the backup and to name and restrict the size of the backup file. You are also prompted to indicate whether you want to run a full or incremental backup. (An incremental backup backs up all changes that have occurred to the directory since the last full or incremental backup.)

Just before you click the Start button to start a backup, you have one last prompt, which, unlike the others, is worth expounding on. The final prompt gives you the option to specify files that are not part of the eDirectory database but that you want to include in the backup file. Assuming that you want to use your backup file to completely restore eDirectory, you should back up some additional files.

For example, you should click to select the option to back up Novell International Cryptographic Infrastructure (NICI) files. NICI files contain cryptographic information that the database uses to decrypt encrypted information and to establish Secure Sockets Layer (SSL) connections. "If you had something encrypted" in your eDirectory tree, Hawkins points out, "and you didn't have NICI files, your backup files would be useless: You couldn't get to that encrypted information." Hawkins included the option to back up additional files expressly for this purpose: to enable you to back up all of the files you need to completely restore your eDirectory database.

Backup eMTool saves the backup file to a location on the server that you specify (for example, `sys://system/backup.bak`). In other words, Backup eMTool does not save the eDirectory backup to storage media. Therefore, you should back up your file system to storage media after you back up your eDirectory

database. This way, you will be able to access your backup files in the event that the server crashes.

When a server does crash and you want to restore the database on that server, you first restore the full backup and then restore the incremental backups you have run since the last full backup. As a final step, and to ensure you completely restore the eDirectory database, you replay the roll-forward log for that server.

#### THE SECRET TO BACKUP SUCCESS

A roll-forward log continuously records all modifications to the database. That may sound boring, but that record of changes is the secret behind Backup eMTool's ability to restore your database right up to the moment before the server crashed.

For example, suppose you have three servers and you have scheduled full backups for Friday nights and incremental backups on Monday through Thursday nights. Now suppose that one of your servers crashes on a Wednesday afternoon, just before you run the incremental backup.

To restore your database on the downed server, you would first restore the full backup and then restore the incremental backups you had run since the full backup. (In this case, you would restore the backup you ran Monday night and then restore the backup you ran Tuesday night.) Backup eMTool prompts you for these incremental backups, asking you to enter their sequence numbers, which you can find in the headers of these backup files. (For more information about these sequence numbers, see "Understanding the Headers for Your Backups.")

Although completing these steps will ensure that nearly all of the eDirectory database is recovered, these two steps alone cannot restore the database right up to the moment the server crashed. After all, changes were made to the database all day Wednesday. If the restored server isn't current with the other servers, this is a problem, particularly if this server shares a replica with one or both of the other servers.

This is where the roll-forward log comes into play. After restoring the full backup and the incremental backups, you replay the roll-forward log for this server to restore all of the modifications to the directory that have occurred since the last

## Understanding the Headers for Your Backups

How will you know which roll-forward log to replay? You simply check the header in your last backup file. Backup files, which are created in binary code, are prefaced with headers, which are displayed in eXtensible Markup Language (XML) format. This XML-formatted header is machine parsable and humanly readable and includes "a gob of information," according to Novell engineer Brian Hawkins.

You can open the backup file in Notepad and read this gob of information to learn all sorts of useful tidbits about a

particular backup file. For example, the header reveals the following information:

- The type of backup (incremental or full)
- The start time of the backup
- The server from which this backup was made
- The operating system running on this server
- The version of eDirectory running on this server
- The roll-forward log that was used when the backup was made

This log will be the starting point for restoring your database. ●

incremental backup. In this case, you replay the roll-forward log to capture any changes made since you backed up this server on Tuesday night. And you're done. Your database is completely restored, and to ensure the database is up-to-date, Backup eMTool then runs a verification check.

#### BACK UP! WHAT ABOUT CONFIGURING BACKUPS?

By default, roll-forward logs are turned off because of the associated management overhead. "When roll-forward logs are turned on," Hawkins explains, "they grow until they fill up

Please visit our advertiser  
*Beginfinite* at  
[www.beginfinite.com](http://www.beginfinite.com).

Please visit our advertiser  
*DSI Consulting Inc.* at  
[www.dsi-consulting.com](http://www.dsi-consulting.com).

whatever volume they're on . . . and when this volume fills up, the database will stop allowing transactions to occur."

Consequently, turning on roll-forward logs requires careful monitoring of the volume on which the roll-forward logs are stored. Ideally, says Hawkins, you should set up a volume specifically for roll-forward logs so you can more easily monitor the size of that volume. When you run a full backup, you can back up the roll-forward logs in this volume and then delete these logs. "In the future," Hawkins says, "we'll make a tool to automate this process, but that's the way it stands for now and that's why roll-forward logging is turned off by default."

You can turn on and otherwise configure roll-forward logs by selecting Backup Configuration from the list of eDirectory Maintenance Tools in Novell iManager. In addition to the option to turn on roll-forward logging, backup configuration options enable you to specify the location of the logs and to indicate a maximum and minimum size for the logs.

When a roll-forward log reaches the minimum size, which is 100 MB by default, the database creates a new log file as soon as the current transaction is completed. The maximum size, 4095 MB by default, ensures that this last "current transaction" doesn't cause the log file to grow to some ridiculous size. If (and this is a rare if) the log reaches the maximum size, the eDirectory database backs out of the current transaction, starts a new log file, and reapplies the transaction.

#### IT'S HOT—BUT HOW?

Roll-forward logs account for Backup eMTool's ability to restore your database up to the moment when a server crashes. However, you may still be wondering how you get a consistent view of the server at the moment in time when you started the backup. In other words, suppose you start a backup at 5 p.m. and the backup takes a total of one hour. During this hour, data somewhere in the middle of the database is modified. How does the eDirectory 8.7 database deal with such live modifications?

What actually happens, Hawkins explains, is that the database interrupts the update, copies the original block of data before it is modified, and sends that copy of the data block to a separate database file. The database also flags this data block, indicating that this block was

modified during the backup process and noting the location of the original block of data (prior to modification).

When the backup process reaches this block of modified data, it takes note of the flag, locates the premodified data block, and includes this original data block in the stored backup for this date and time. In this way, the backup of this data block and every other data block in the database is consistent with the time the backup process started. Incidentally, the database engine's ability to capture a consistent view of the database in this way is an improvement to eDirectory 8.7.

#### BACKUPS FOR CATASTROPHIC FAILURE?

Naturally, the primary goal of Backup eMTool is to restore the database on

*A good tree design makes recovery from such catastrophic events easier.*

*A "good" design, in this case, is one that enables you to restore a master replica server and then use synchronization to restore the other servers.*

individual servers. Restoring the database after the sys:// volume fails on a server accounts for as many as 95 percent of all backup and restore cases, Hawkins estimates. Assuming you have moved your roll-forward logs to a volume other than the sys:// volume, Backup eMTool works well to restore the database in such cases.

But what about catastrophic cases in which all volumes on your server fail or in which you lose all of your servers to a flood or fire? If all volumes on a single server fail, you can use replication to restore the server (assuming you have more than one tree in your directory). If you lose all of your servers (leaving no replicas), you can use Backup eMTool to

restore your tree—but doing so requires some careful tree planning.

A good tree design makes recovery from such catastrophic events easier. A "good" design, in this case, is one that enables you to restore a master replica server and then use synchronization to restore the other servers.

For example, Hawkins explains, suppose you have a small network with 10 servers and you want to design an eDirectory tree that you could easily restore using only Backup eMTool. In such a case, you could set up one server with a replica of every partition in the eDirectory tree.

By backing up this server every night, you would ensure that you always have a complete view of the entire eDirectory tree. Thus if you lost all of your servers, you would need to restore only that one server, after which you would bring up all of the other servers as new servers in the tree. Finally, you would re-replicate the eDirectory data to the "new" servers and thereby restore your eDirectory database in a minimal amount of time.

Now, suppose you have a more complex network with 100 servers and 100,000 objects. In this case, you could set up three servers, each of which contains one-third of the tree and do not share replicas. Again, you would back up these servers every night. If you lost your entire data center, you would restore these three servers and thereby restore your entire eDirectory tree.

Hawkins admits that using such a method might incur a few inconsistencies. However, the cost of a few inconsistencies might be worth the reward of restoring your eDirectory database with such relative ease.

#### PAY ATTENTION

If you have a relatively large database, Backup eMTool should sound pretty good: If it doesn't sound pretty good, perhaps you're not paying attention.

Prior to Backup eMTool, backing up eDirectory meant backing up objects one at a time, which meant backups were painfully slow. Although the old backup method still has its uses, says Hawkins, the new method more effectively and far more efficiently solves the problem of restoring eDirectory when servers go down.

Linda Kennard works for Niche Associates, which is located in Sandy, Utah. Brian Hawkins is a Novell engineer who wrote Backup eMTool. ●

Please visit our advertiser *Novell Inc.* at  
[www.novell.com](http://www.novell.com).