



ADMINISTRATOR ESSENTIALS

Backup methods for Domino

By [Ron Herardian](#)

Backing up Domino is often a challenge. Getting good backups is not just a matter of slapping a DAT on each server machine. In this article, I'll look at the problem of backups from a low-level technical perspective, then go into some system design considerations for backup and restoration of Domino servers.

The basic problem with backing up Domino databases (and this is often the case with many server-based systems besides Domino) is backing up open files. This is an issue for every database system and for every major backup system. There are four basic options for backing up a database and they apply to most database systems as well as to Domino. Database administrators usually understand the problem better than Domino administrators, but with this article, you'll be completely clued in.

Here's an interesting hint for you cc:Mail types: Remember that cc:Mail is a database system much more than it is a data communications system.

The two simplest ways to handle open database files are to either (a) shutdown and backup or (b) to do a brute force read of the database without shutting down. A much better approach is to use a backup system that can handle open files. ArcServe, for example, is one product that supports backing up of various database systems. Most network backup systems support backup of open files or provide agents designed for compatibility with specific database systems.

How to backup open files

There are four basic ways to backup open files if you want everything to work successfully. They are linear read, the use of open file managers, the use of clustering, and our old favorite: shutdown and backup. There's also the network backup and restore approach, but that's not going to be a win on WANs. Read on for details.

Linear read

Most copy and backup utilities assume that a file that is already open shouldn't be copied or backed up because there's no way to guarantee the integrity of data within a single file or across files, within a set of related files. However, there are backup programs and utilities that allow "hot" copies of live database files. This is a most crude backup method and it brings with it two basic limitations.

It's technically possible to simply read open files linearly without regard to ongoing changes (this works if you open the files in sharing mode). The problem with this approach is that database files in the backup may be inconsistent both internally and with each other (in a set of related files) due to ongoing changes during the backup. In other words, the backup process can open and read a file that is being modified, but, as the backup proceeds through the file, later portions of the file may be inconsistent with earlier portions due to live modifications. Also, when a set of files contains references across files, the file set may be inconsistent across files when each file is read in turn -- while all of the files are being modified.

Depending on the value and the nature of the data as well as on the availability of tools to reconcile differences within and between files, the linear read backup method may be acceptable. However, most database systems contain critical data and low-level database analysis and repair tools are often unavailable or limited. Oracle, for example, is not tolerant such inconsistencies resulting from hot backups and may not be repairable upon restore. The accepted method is to fall back to a known good version of the database and reapply logged transactions to

produce a reconstructed version of the data up to the minute of a failure. Of course, if you don't have the ability to roll in logged transactions, your database is toast.

Open file managers

The second way to backup open files is through true open file management in the backup system. This means that the backup system will wait to start the backup of a given file until all write operations have completed. Once there are no writes pending, the backup will take a snapshot of the file system data for that file and begin backing up. Systems such as Veritas provide this type of advanced functionality.

In this case, as changes to the open file continue during the backup, the open file manager intercepts write calls to the file being backed up, and stores any changed records or sectors of the file in a temporary location. As the backup proceeds, any changed records are read from the temporary backup so that the backup system ultimately produces a copy of the open file exactly as it was when no writes were pending. This method ensures internal file consistency but not consistency across files in a set of related files. When using this type of backup technology, it is still necessary to run repair and maintenance procedures before putting a restored database back online. However, this method is adequate for most Domino databases.

More sophisticated agents are available from some backup vendors that are capable of ensuring integrity across files by applying the same open file management logic, at one time, to a complete set of files. In this case the backup software, in effect, takes a 'snapshot' of the file system, then logs changes as it proceeds to backup the file system, fetching data from a temporary cache for any files that are modified during the backup. On a busy server, however, this may require a brief shutdown of server processes to bring the file system to a known state before starting the backup (this functionality is available with some versions of the HP/UX operating system).

Clustering

A more sophisticated approach is to use clustering (e.g., NetWare SFT 3 or Domino clustering), then break the cluster and backup the secondary server (and when finished, reestablish the cluster). While this is a technically excellent solution that eliminates a variety of problems, it requires a cluster of servers -- which means at least one additional server machine with identical storage resources.

A refinement to this solution for Domino is to archive data to a cluster server maintaining a lesser amount of data on a primary server (this can be done through Domino replication and server-based agents). This makes it possible to bring the primary server back on line quickly since it will have smaller databases to process.

Although clustering is being recommend here as a backup and archiving solution the larger benefit is high availability (H/A) and fail-over should the primary server fail.

Shut down and backup

The final, and simplest, method is well known to many administrators. Domino administrators often automate server shutdown and backup with scripts such as (for Windows NT Server):

```
net stop lotus notes server
<backup command here>
net start lotus notes server
```

Network backup and restore

Large systems often demand network backups using tools like NetBackup, Enterprise ArcServe, and IBM's ADSM. However, while these systems work well on high-speed server backbones they do not work well over a WAN. More to the point, restore of several gigabytes of data over typical WAN link is unacceptably slow.

Summary

Backing up Domino presents a set of technology issues and backup requirements similar to other database systems. The crude solution is to shut down the server, back up, and bring the server back online -- but this is unacceptable where 7x24 up time is required (and since Domino is a Web server, 24/7 is the expectation). The most elegant solutions are sophisticated backup software capable of open file management and an investment in additional hardware for clustering.

Product availability and resources

Enterprise ArcServe is available from Cheyenne Software at <http://www.cai.com/arcserveit/>.

Veritas NetBackup is available at <http://www.veritas.com>.

IBM's ADSM <http://www.storage.ibm.com/software/adsm/>.

Ron Herardian is CEO & Chief Systems Architect at Global System Services (GSS). You can reach Ron via email at rherardi@gssnet.com, or via his web page at <http://www.gssnet.com>.

Copyright © 1998-2008, [ZATZ Publishing](#). All rights reserved worldwide.