Improving the Performance of Servers by Implementing the Manual Log Shipping for Large Databases

Anil[1], Parveen Yadav[2], Balraj Sharma[3], Vinit Lohan[4]

[1]M.Tech C.S.E, BITS, Bhiwani, Haryana-127021, India
anilverma.bits@gmail.com

[2]Assistant Professor, BITS, Bhiwani Haryana-127021, India
par.biran@gmail.com

[3]Assistant Professor, BITS Bhiwani, Haryana-127021, India
balrajbhardwas@gmail.com

[4]Assistant Professor, VCE, Rohtak-124001, India
lohanvinit@gmail.com

Abstract

Business success, the need to ensure 24x7availability is greater than it has ever been. One common method for as database systems become more and more critical to providing 99.99%, or "4-nines" availability, is to implement a "warm" standby database server. The use of standby servers is not a new concept; database administrators (DBAs) have been using them for years. Typically, standby servers require the DBA or operator to create backups of the database and logs on the primary system and then restore these backups on a standby server on a regular basis. If the primary server fails, the down time is isolated to the amount of time required to process the log files since the last backup was restored on the standby server.

Log shipping allows you to automatically send transaction log backups from a primary database on a primary server instance to one or more secondary databases on separate secondary server instances. The transaction log backups are applied to each of the Secondary databases individually. An optional third server instance is known as the monitor server records. The history and status of backup and restores operations and, optionally, raises alerts if these operations fail to occur as scheduled.

We can improve the functionality of our Log Shipping manually so that back up is made quickly and making it quick, we insert some manual codes. Implementing Log Shipping for small databases is relatively straightforward as one can easily do it using the GUI but for large database, the GUI is not a good option as the database on the destination server will be always out of sync. So a question arises *how do we accomplish this task?* In this paper, we are going to explain you the detailed steps which we followed to implement the log shipping for our large databases manually and improving the performance of our servers.

Keywords: *Replication, backup, recovery, hot backups, online backups, dynamic changes, reconciliation.*

1. Introduction

For distributed database application environment, it is always required to synchronize different database servers, back up, copy Transaction **Log**s, etc. If we are going to implement using application we have to put lots of efforts to build up the application. SQL Server 2005 provides an advanced feature called **Log Shipping**. **Log shipping** is an Automated Process for backing up, restoring, copying the transaction **log**s and synchronizing the database for distributed database server application which can improve the application performance and availability of database.

Log Shipping is used to synchronize the Distributed Database Server. Synchronize the database by copying Transaction **log**s, Backing up, Restoring data. SQL Server used **SQL Server Job Agents** for making those processes automatic. **Log Shipping** does not involve automatic.

Transfer of server if there is any failure. This means it has just synchronized the databases but if the primary server fails, it will not redirect your application to a secondary server. This has to be done manually.

The main functions of **Log Shipping** are as follows:

- Backing up the transaction **log** of the primary database
- Copying the transaction **log** backup to each secondary server
- Restoring the transaction **log** backup on the secondary database

The figure 1 shows a log shipping configuration with the primary server instance, three secondary servers.

Instances and a monitor server instance. The figure illustrates the steps performed by backup, copy, and restore jobs, as follows:

1. The primary server instance runs the backup job to back up the transaction log on the primary database. This server instance then places the log backup into a primary log-backup file, which it sends to the backup folder. In this figure, the backup folder is on a shared directory—the *backup share*.
2. Each of the three secondary server instances runs its own copy job to copy the primary log-backup file to its own local destination folder.

3. Each secondary server instance runs its own restore job to restore the log backup from the local destination folder onto the local secondary database.

The primary and secondary server instances send their own history and status to the monitor server instance.

2. Terms and Definitions and Architecture

2.1 Log shipping

Log shipping is an Automated Process for backing up, restoring, copying the transaction **log**s and synchronizing the database for distributed database server application which can improve the application performance and availability of database.

2.2 Primary server

The instance of SQL Server is a production server.

2.3 Primary database

The database on the primary server is to be back up to another server. All administration of the log shipping configuration through SQL Server Management Studio is performed from the primary database.

2.4 Secondary server

The instance of SQL Server where we want to keep a warm standby copy of your primary database.

2.5 Secondary database

The warm standby copies the primary database. The secondary database may be in either the RECOVERING state or the STANDBY state, which leaves the database available for limited read-only access.

2.6 Monitor server

An optional instance of SQL Server that tracks all of the details of log shipping, including:

- When the transaction log on the primary database was last backed up.
- When the secondary servers last copied and restored the backup files.
- Information about any backup failure alerts.

2.7 Backup job

A SQL Server Agent job that performs the backup operation, logs history to the local server and the monitor server, and deletes old backup files and history information. When log shipping is enabled, the job category "Log Shipping Backup" is created on the primary server instance.

2.8 Copy job

A SQL Server Agent job that copies the backup files from the primary server to a configurable destination on the secondary server and logs history on the secondary server and the monitor server. When log shipping is enabled on a database, the job category "Log Shipping Copy" is created on each secondary server in a log shipping configuration.

2.9 Restore job

A SQL Server Agent job that restores the copied backup files to the secondary databases. It logs history on the local server and the monitor server, and deletes old files and old history information. When log shipping is enabled on a database, the job category "Log Shipping Restore" is created on the secondary server instance.

2.10 Alert job

A SQL Server Agent job that raises alerts for primary and secondary databases when a backup or restore operation does not complete successfully within a specified threshold. When log shipping is enabled on a database, job category "Log Shipping Alert" is created on the monitor server instance.

The following figure shows a log shipping configuration with the primary server instance, three secondary server instances, and a monitor server instance. The figure illustrates the steps performed by backup, copy, and restore jobs, as follows:

1. The primary server instance runs the backup job to back up the transaction log on the primary database. This server instance then places the log backup into a primary log-backup file, which it sends to the backup folder. In this figure, the backup folder is on a shared directory—the backup share.
2. Each of the three secondary server instances runs its own copy job to copy the primary log-backup file to its own local destination folder.
3. Each secondary server instance runs its own restore job to restore the log backup from the local destination folder onto the local secondary database.

The primary and secondary server instances send their own history and status to the monitor server instance.

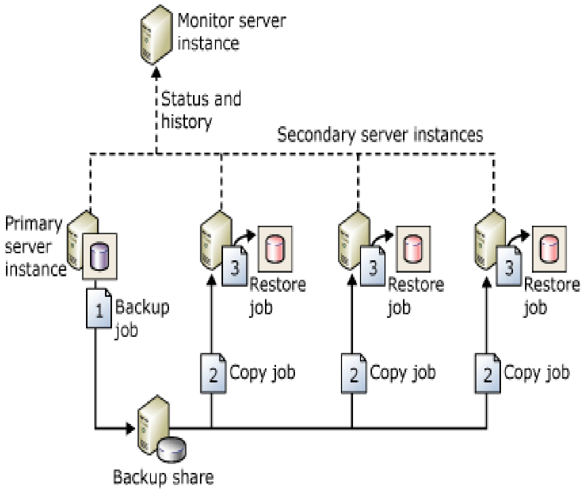


Fig. 1 Sequential Log Shipping

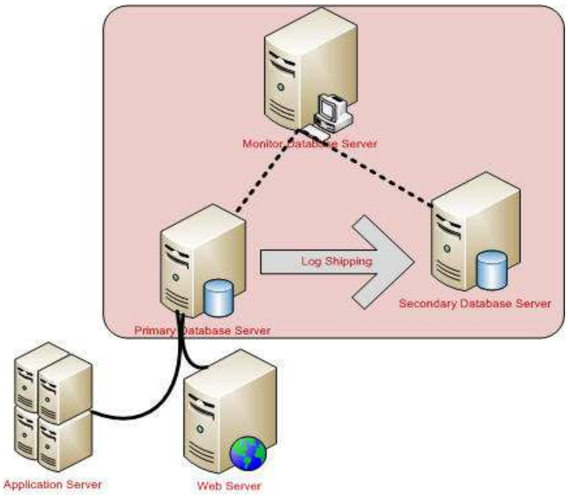


Fig 2 Automatic Log Shipping

3. Database Mirroring and Log Shipping (SQL Server) & manually Log shipping

A given database can be mirrored or log shipped; it can also be simultaneously mirrored and log shipped. To choose what approach to use, consider the following:

- How many destination servers do you require?

If you require only a single destination database, database mirroring is the recommended solution.

If you require more than one destination database, you need to use log shipping, either alone or with database mirroring. Combining these approaches gives you the benefits of database mirroring along with the support for multiple destinations provided by log shipping.

- If you need to delay restoring log on the destination database (typically, to protect against logical errors), use log shipping, alone or with database mirroring.

This topic discusses considerations for combining log shipping and database mirroring.

The principal database in a mirroring session can also act as the primary database in a log shipping configuration, or vice versa, as the log shipping backup share is intact. The database mirroring session run in any operating mode, whether synchronous (with transaction safety set to FULL) or asynchronous (with transaction safety set to OFF).

Typically, when combining log shipping and database mirroring, the mirroring session is established before log shipping, although this is not required. Then the current principal database is configured as the log shipping primary (the principal/primary database), along with one or more remote secondary databases. Also, the mirror database must be configured as a log shipping primary (the mirror/primary database). The log shipping secondary databases should be on different server instances than either the principal/primary server or mirror/primary server.

During a log shipping session, backup jobs on the primary database create log backups in a backup folder. From there, the backups are copied by the copy jobs of the secondary servers. For the backup jobs and copy jobs to succeed, they must have access to the log shipping backup folder. To maximize availability of the primary server, we recommend that you establish the backup folder in a shared backup location on a separate host computer. Ensure that all the log shipping servers, including the mirror/primary server, can access the shared backup location (known as a backup share).

To allow log shipping to continue after database mirroring fails over, you must also configure the mirror server as a primary server, using the same configuration you use for the primary on the principal database. The mirror database is in the restoring state, which prevents the backup jobs from backing up the log on the mirror database. This ensures that the mirror/primary database does not interfere with the principal/primary database whose log backups are currently being copied by secondary servers. To prevent spurious alerts, after the backup job executes on the mirror/primary database, the backup job logs a message to the l**og.**

Shipping monitor history detail table, and the agent job returns a status of success.

The mirror/primary database is inactive in the log shipping session. However, if mirroring fails over, the former mirror database comes online as the principal database. At that point, that database also becomes active as the log shipping primary database. The log shipping backup jobs that were previously unable to ship log on that database, begin shipping log. Conversely, a failover causes the former principal/primary database to become the new mirror/primary database and enter the restoring state, and backup jobs on that database cease to backup log.

To run in high-safety mode with automatic failover the mirroring session is configured with an additional server instance known as the witness. If the principal database is lost for any reason after the database is synchronized and if the mirror server and witness can still communicate with each other, automatic failover occurs. An automatic failover causes mirror server to assume the principal role and bring its database online as the principal database. For more information, see Automatic Failover (Database Mirroring). If the log shipping backup location is accessible to the new principal/primary server, its backup jobs begin to ship log backups to that location. The database mirroring synchronous mode guarantees that the log chain is unaffected by a mirroring failover and that only valid log is restored. The secondary servers continue to copy log backups without knowing that a different server instance has become the primary server.

When using a local log shipping monitor, no special considerations are necessary to accommodate this scenario. For information about using a remote monitoring instance with this scenario, see, "The Impact of Database Mirroring on a Remote Monitoring Instance," later in this topic.

3.1 Failing over from the Principal to the Mirror Database

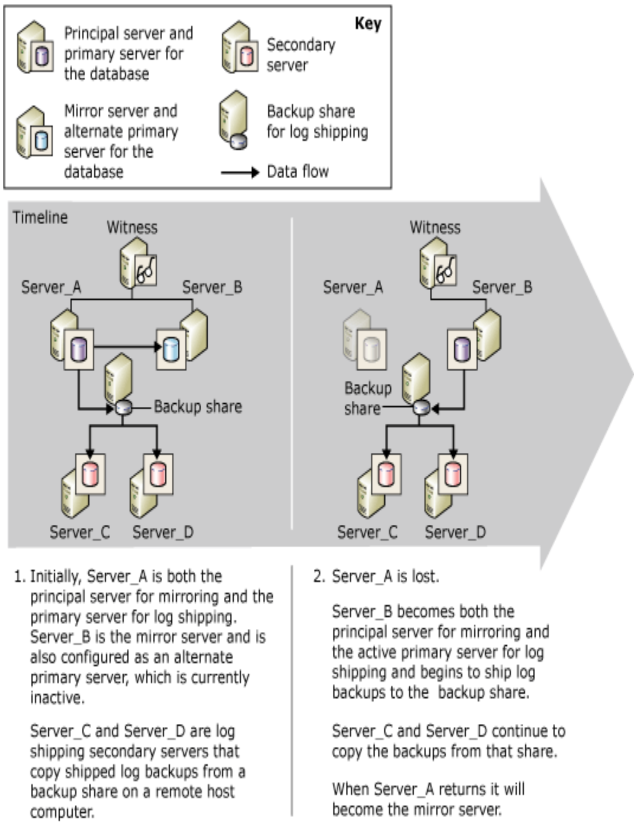
The following figure shows how log shipping and database mirroring work together when mirroring is running in high-safety mode with automatic failover. Initially, **Server A** is both the principal server for mirroring and the primary server for log shipping. **Server B** is the mirror server and is also configured as a primary server, which is currently inactive. **Server C** and **Server D** are log shipping secondary servers. To maximize availability of the log shipping session, the backup location is on a share directory on a separate host computer.

After a mirroring failover, the primary server name defined on the secondary server is unchanged.

3.2 The Impact of Database Mirroring on a Remote Monitoring Instance

When log shipping uses with a remote monitoring instance, combining the log shipping session and database mirroring affects the information in the monitor tables. The information about the primary is a combination of the one configured at the principal/primary and the monitor configured on each secondary.

To keep monitoring as seamless as possible, when you use a remote monitor, we recommend that you specify the original primary name when configuring the primary at the secondary. This approach also facilitates changing the log shipping configuration from Microsoft SQL Server Agent.


Key
Principal server and primary server for the database
Secondary server
Mirror server and alternate primary server for the database
Backup share for log shipping
Data flow
Timeline
Witness
Server_A
Server_B
Backup share
Server_C
Server_D
Witness
Server_A
Server_B
Backup share
Server_C
Server_D

1. Initially, Server_A is both the principal server for mirroring and the primary server for log shipping. Server_B is the mirror server and is also configured as an alternate primary server, which is currently inactive.

Server_C and Server_D are log shipping secondary servers that copy shipped log backups from a backup share on a remote host computer.

2. Server_A is lost.

Server_B becomes both the principal server for mirroring and the active primary server for log shipping and begins to ship log backups to the backup share.

Server_C and Server_D continue to copy the backups from that share.

When Server_A returns it will become the mirror server.

Fig 3 Mirroring Failure

For more information about monitoring, see Monitor Log Shipping (Transact-SQL).

Setting Up Mirroring and Log Shipping Together

To set up database mirroring and log shipping together, the following steps are required:

1. Restore backups of the principal/primary database with NORECOVERY onto another server instance to be later used as database mirroring mirror database for the principal/primary database. For more information, see Prepare a Mirror Database for Mirroring (SQL Server).
2. Set up database mirroring. For more information, see Establish a Database Mirroring Session Using Windows Authentication (SQL Server Management Studio) or Setting up Database Mirroring (SQL Server).
3. Restore backups of the principal/primary database to other server instances to be later used as log shipping secondary databases for the primary database.
4. Set up log shipping on the principal database as the primary database for one or more secondary databases.

You should set up a single share as the backup directory (a backup share). This ensures that after role switching between the principal and mirror servers, backup jobs continue to write to the same directory as before. A best practice is to ensure that this share is located on a different physical server from the servers hosting the databases involved in mirroring and log shipping.

5. Manually failover from the principal to the mirror.

To perform a manual failover:

- Manually Fail Over a Database Mirroring Session (SQL Server Management Studio)
- Manually Fail Over a Database Mirroring Session (Transact-SQL)

6. Set up log shipping on the new principal (previously mirror) as the primary database.
7. You must use the same backup share that you used in step 4.
8. The Transaction Log Shipping interface in SQL Server Management Studio supports only one primary database per log shipping configuration. Therefore, you must use stored procedures to set up the new principal as primary.
9. Perform another manual failover to fail back to the original principal.

4. Advantage

- Provides a disaster-recovery solution for a single primary database and one or more secondary databases, each on a separate instance of SQL Server.
- Supports limited read-only access to secondary databases (during the interval between restore jobs).
- Allows a user-specified delay between when the primary server backs up the log of the primary database and when the secondary servers must restore (apply) the log backup. A longer delay can be useful, for example, if data is accidentally changed on the primary database. If the accidental change is noticed quickly, a delay can let you retrieve still unchanged data from a secondary database before the change is reflected there.

5. Disadvantages of Log Shipping

Unfortunately, log shipping does have a few issues that the DBA should be aware of prior to relying on this method of failover:

- Possible data loss when the primary server fails. If the primary server becomes completely unusable, transactions that occurred after the last transaction log backup that was copied to the standby server are lost. For example, suppose that server A fails at 5 a.m. and you cannot get to it at all. If the last backup copied to server B was taken at 4:45 a.m., all transactions that occurred between 4:45 a.m. and 5 a.m. are lost forever.
- Some manual DBA work is required to bring the standby server online, as discussed in this article.
- Log shipping setup cannot be scripted. This means that you cannot mimic the production environment for testing purposes without going through the wizard screens.
- The Enterprise edition of SQL Server 2000 is required on primary and standby servers. If you run any other version/edition of SQL Server, you're out of luck. The Developer edition can be used to learn how to set up log shipping, but it cannot be used in a production environment. Note however, that log shipping is merely an automated way of copying transaction log backups, so a savvy DBA can easily set up jobs to accomplish the same functionality.

Difficult troubleshooting Log shipping usually works very well, but if there are problems, they're difficult to troubleshoot—documentation is sparse and typically not helpful for solving a particular problem. Fortunately Microsoft's Knowledge Base articles have good information for troubleshooting log-shipping issues.

Each database that needs to be log-shipped must be set up through a separate maintenance plan.

6. Conclusions

The objective of this work is to identify the possibilities in the technology and to write an algorithm to make the log shipping from production database at one location to the standby database at remote geographic location fastest.

The approach behind this research will be two pronged and compressing the redo logs to the optimum in advance. Doing the shipping of compressed logs parallel to the remote standby database location and uncompressing them to the original forms so that they can be applied on the standby database.

References

[1] Microsoft SQL Server 2008 R2 Unleashed by Ray Rankins, Paul T. Bertucci, Chris Gallelli and Alex T. Silverstein.

[2] Inside SQL Server book

[3] High Availability with SQL Server 2008

[4] Beginning SQL Joes 2 Pros: The SQL Hands-On Guide for Beginners (SQL Exam Prep Series 70-433 Volume 1 of 5) (Sql Design Series) by Rick A Morelan

[5] Microsoft SQL Server 2008 Reporting Services by Brian Larson