

---

*Project Name: Implementing Log shipping between database servers*  
*Company: Nexterna Inc.*

---

Scope:

This document provides process guidelines used to Design & Implement Log Shipping between two database servers and one monitor server.

The issues identified needs repetitive testing and analysis before implemented in production environment. The process involves knowledge SQL Server 2000 Enterprise Mgr and T-SQL commands.

Process Overview:

Log Shipping allows you to automatically send transaction log backups from one database (known as the primary database) to a secondary database on another server (known as the secondary server). At the secondary server, these transaction log backups are restored to the secondary database, keeping it closely synchronized with the primary database. An optional third server 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.

Specifications:

Primary Server	: Windows 2000 OS - SQL Server 2000 SP4
Secondary Server	: Windows 2000 OS - SQL Server 2000 SP4
Monitor Server	: Windows 2000 OS - SQL Server 2000 SP4
Persons Involved	: (1) Database Administrator (2) System Administrator

Implementation Plan:

Phase I	: Setting up Log Shipping
Phase II	: Monitoring Log Shipping
Phase III	: Role Change & Role Reversal

Process Details:

Setting up Log Shipping

The primary server is the production server which holds the source database. The secondary server holds the destination database to which you copy and restore the source database's transaction logs. A monitor server is used to monitor primary and secondary servers participating in Log shipping operation.

Before you start the process of setting up Log shipping, identify the following steps:

- Identify the primary, secondary and monitor servers. Make sure that the SQL Server installation on all these servers is identical and proper service packs are applied to it.
- Though monitor server is optional, Microsoft recommends a separate server.
- Establish security to all the servers. The windows account you use to setup log shipping must have SQL Server System Administrator (sa) privileges on all the servers.

- Create primary and secondary file shares. First, create a share for the folder in which the source database's transaction logs will reside. Second, create a share for the secondary-server folder to which you plan to copy and restore the transaction-log files. To be more legitimate, specify the server and database names in the file share. Grant permissions on these file shares to the Windows account that each server's SQL Agent is using.
- Decide how to create and synchronize the destination database. You can let the log shipping setup process create this for you.
- Register all the three servers in Enterprise Manager

After we complete these initial preparations, we can use Database Maintenance plan wizard of Enterprise Manager to setup log shipping. The process is done in five steps:

1. One time backup of source database
2. Copy backup & restore to the secondary server
3. Backup Transaction logs to Disk file.
4. Copy the transaction log files to secondary server
5. Restore the transaction logs to the destination database.

### Monitoring Log Shipping

After log shipping is in place, SQL Server enables Enterprise Manager's Log shipping monitor utility on the monitor server. In addition, SQL Server also creates two SQL Agent alert jobs: one for backup and the other for out-of-sync conditions.

Drill down the Management node in Enterprise Manager Console and select the Log shipping monitor. It presents a list of log shipping pairs. The copy and restore histories give you quick access to more detailed error information. This monitor contains several other things which give you overview of the log shipping process.

### Role Change & Role Reversal

Role reversal is a process of role change in which, you recover the secondary server database and designate it as the new primary server database in event when production database of primary server is out of service.

This process of role change is divided into six basic steps:

1. Transferring and exporting logins
2. Demoting the primary server
3. Promoting the secondary server
4. Informing the monitor server of the role change
5. Resolving the logins on the secondary server
6. Linking database access to permissions

#### 1. Transferring and exporting logins

First, build DTS package to transfer logins from the primary server to the secondary server and resolve login SIDs across distinct servers. You create and save the DTS package on the primary server, then setup the package execution by invoking dtsrun.exe through a SQL Server Agent job on the primary server. The package execution transfers the logins from one server to the other, but it doesn't resolve their login SIDs. However, to be able to resolve login IDs later, you must first create a file containing an export of the primary server's syslogins table. To export the logins to the secondary server, you create a two-stage SQL Server Agent job: bcp out and copy. In the first step, you export the logins to a file by using bcp in native mode. In the second step, you copy the logins to a file on the secondary server that you can use later for

resolving logins during the role change. At that point, you use `sp_resolve_logins` stored procedure to resolve login SIDs on the secondary server. After you create the job, you can run it at regular intervals to keep an up-to-date exported file of logins on the secondary server in case you need to make a log shipping role change.

## 2. Demoting the primary server

To take the primary server out of its role as the source of log shipping, you “demote” it to a lesser position.

You can demote the primary server source database from a production server to a potential secondary server and remove it from log shipping by executing the `sp_change_primary_role` stored procedure on the primary server. See example;

```
Use master
GO
EXEC msdb.dbo.sp_change_primary_role
    @db_name = 'Pubscopy',
    @backup_log = 1,
    @terminate = 1,
    @final_state = 3,
    @access_level = 1
```

The above script removes the primary server database from the log shipping plan. The parameters tell the stored procedure to make one last transaction-log backup, terminate all users in the database, then put the database into a standby final state and into a multiuser access level. The stored procedure’s return code states whether the BACKUP LOG statement succeeded.

## 3. Promoting the secondary server

The next step is to promote the current secondary server database to a recovered state so that it can be used in place of the original production database and can become a possible primary log shipping database. On the secondary server, after you make sure no other users are accessing the database, you can execute the `sp_change_secondary_role` stored procedure as shown below;

```
USE master
GO
EXEC msdb.dbo.sp_change_secondary_role
    @db_name = 'Pubscopy',
    @db_load = 1,
    @force_load = 1,
    @final_state = 1,
    @access_level = 1,
    @terminate = 1,
    @keep_replication = 0,
    @stopat = null
```

The parameters cause the stored procedure to attempt to copy all remaining log files from the former primary server and load all the remaining copied transaction logs to the secondary server. Passing `@do_load = 1` parameter makes a last copy and load of all remaining transaction logs, and passing the `@force_load = 1` parameter specifies the undocumented `forceload` option on `sqlmaint.exe`. The `@final_state = 1` parameter puts the new primary database in recovery mode, and the `@access_level` parameter sets the access back to multiuser. The `@terminate = 1`

parameter causes the stored procedure to terminate all users accessing the database by issuing the ALTER DATABASE command with the ROLLBACK IMMEDIATE option. Last, if the database is a publishing replication database, the @keep\_replication = 0 parameter will maintain the server's replication settings.

When the stored procedure finishes and brings the database online, it will send a message stating that the RESTORE DATABASE command succeeded. After this jobs starts, transaction-log backup files will start appearing on the new primary server. You might need these files to establish log shipping back to the new secondary server.

#### 4. Informing the monitor server of the role change

SQL Server 2000 log shipping installs a monitoring utility on a monitor server, preferably a third server. To notify the monitor server of the role change, you now need to execute on the monitor server the sp\_change\_monitor\_role stored procedure as shown below;

```
USE master
GO
EXEC msdb.dbo.sp_change_monitor_role
    @primary_server = 'HSHAH\Server1',
    @secondary_server = 'HSHAH\Server2',
    @database = 'Pubscopy',
    @new_source = 'HSHAH\Server2'
```

Despite its name, the stored procedure doesn't change the monitor's role. Instead, stored procedure changes the references to secondary and primary server file shares. That is, it deletes rows in the monitor server's log\_shipping\_secondaries table that referenced the old secondary server, then replaces the old primary server name with the new primary server name in the log\_shipping primaries table. The stored procedure doesn't insert rows into the log\_shipping\_secondaries table because a log shipping pair doesn't yet exist.

#### 5. Resolving the logins on the secondary server

Resolving the old primary server logins on the new primary server lets users access the new primary server. You can resolve the logins on the new production server by using the logins file that you exported in step 1. the sp\_resolve\_logins stored procedure reads the exported logins file, then resolves the differing SIDs between the servers. See example below;

```
USE master
GO
EXEC sp_resolve_logins
    @dest_db = 'Pubscopy',
    @dest_path = 'd:\',
    @filename = 'syslogins.dat'
```

The above example shows how you can execute the sp\_resolve\_logins stored procedure to resolve logins on the newly recovered Pubscopy database. Run this stored procedure from the master database.

#### 6. Linking database access to permissions

To link the transferred and resolved logins with their corresponding database users and permissions, in the new primary server's production database, you need to execute the sp\_change\_users\_login stored procedure once for each login:

```
USE Pubscopy
GO
EXEC sp_change_users_login 'update_one', 'username', 'loginname'
```

Executing this stored procedure ensures that the SQL Server logins link correctly to their corresponding usernames in the database.

At this point, we have successfully promoted the secondary server to its new role, and the old primary server is ready to become a secondary server. However, you we still have not established a log shipping relationship. We have made a role change but not a role reversal.

#### Role Reversal:

To accomplish we need to setup log shipping from the new primary server to the new secondary server. Because the new primary server contains a new database maintenance plan, our natural inclination is to add the new secondary server as a destination server in that plan. However, after repeatedly trying to add the new secondary server as a destination server, we found that the transaction-log backup job on the new primary server fails, and log shipping won't start from the new primary server to the new secondary server.

After we apply the log shipping role-change stored procedures and tasks that detailed earlier, we can complete a full role reversal by setting up a new log shipping plan from the new primary server to the old secondary server. We followed the following steps to setup the plan:

- Remove log shipping from the database maintenance plan on the new primary server
- Delete the database maintenance plan on the primary server
- Delete the database maintenance plan on the secondary server
- Retain all transaction log files
- Create a new database maintenance plan on the new primary server, specifying the new secondary server and database and appropriate transaction-log file locations.
- Resume application activity on the new primary server.

If log shipping monitor returned to error, wait until the load job loads the latest backup file on secondary server and the log shipping monitor will return to error free-state.

#### Conclusion:

Setting up and monitoring log shipping can be done thru enterprise manager. However, making role changes and role reversals requires extra work. Enterprise manager does not have ability to make log shipping role change and role reversals. You need to manually apply stored procedures to accomplish these tasks.

#### References:

- SQL Server 2000 Books online
- Microsoft TechNet