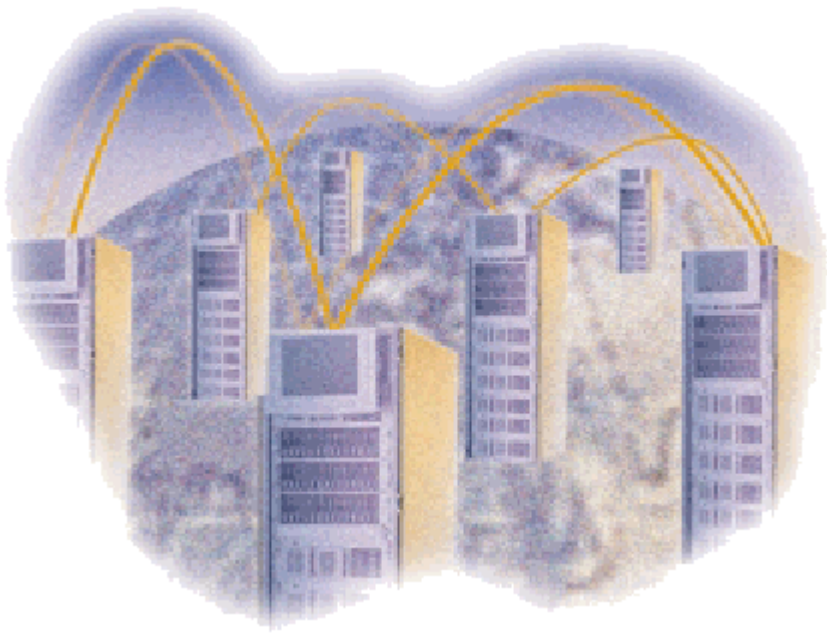




## Backup Methodologies Using Double-Take 4.x



Backup Methodologies Using Double-Take 4.x published October 2001

NSI and Double-Take are registered trademarks of Network Specialists Inc. All other products are trademarks of their respective companies. © 1996–2001 NSI Software

## *Double-Take Support for Application Failover*

Double-Take's file system replication process is application independent and replicates any file system changes (including permissions and attributes) written to NTFS, FAT or FAT32 file systems by any application or process, subject to specific exceptions called out in the *User's Guide* or *readme.txt* file. Maintaining point-in-time consistent file system replicas and providing server monitoring and automatic or manual failover of the server name and IP address are the primary functions of the Double-Take software and we offer support to qualified customers should these functions fail to operate in accordance with our published documentation, regardless of what application or process is manipulating the data.

NSI Software may provide application notes and other documents that provide implementation guidelines on how to use Double-Take functions and replicas to manually or automatically failover or recover many popular third party applications and a general process to accomplish failover or recovery of many other third party applications. While these steps are believed to be accurate for the specific configuration, Double-Take version, and application versions originally tested, due to the number of possible configurations and variables, NSI Software can only test selected combinations and may provide only limited support for the operation and configuration of third party applications or the behavior of those applications before, during, or after failover, in its discretion. In cases where NSI Software has no direct access to or experience with a particular application or configuration, NSI Software support may also be limited to only the actual replication of the file system data and failover (name and IP address) of the server.

For assistance in validating, implementing or troubleshooting these or other possible configurations with third party applications, NSI Software and its partners may offer professional services on a fee basis to apply best practices for assisting with third party applications to recover automatically or manually using replicated data.

This, and any other, application note is provided solely for the convenience of our customers and is not intended to bind NSI Software to any obligation.

---

# Table of Contents



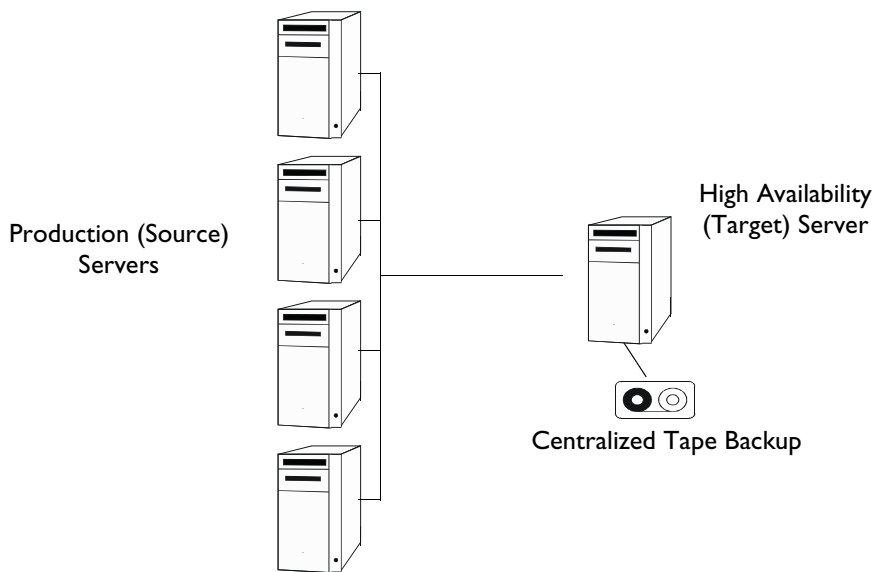
<b>Introduction .....</b>	<b>1</b>
<b>Incremental Backups .....</b>	<b>2</b>
<b>Registry Settings and System Files .....</b>	<b>2</b>
<b>Backing Up Deleted Files .....</b>	<b>2</b>
<b>Applications with Interdependent Files .....</b>	<b>3</b>
<b>Pausing Double-Take .....</b>	<b>4</b>
<b>One-to-many replication .....</b>	<b>6</b>
<b>Disconnecting Double-Take for a backup .....</b>	<b>6</b>
<b>Rotating replicas .....</b>	<b>8</b>

---

# Introduction

The rapid growth in storage brought on by the Internet and distributed computing has placed nearly impossible demands on administrators responsible for protecting corporate data assets. The backup window has shrunk to nearly zero and tape backup systems can introduce significant overhead to a production server, seriously impacting its performance. While the importance of backups increases, the impact of periodic full system backups is obvious. Even nightly incremental backups dominate processing while they examine every file system object and then read all files that have changed in their entirety for backup. Performing this process across a network adds additional overhead as the entire process happens across the wire.

Double-Take can enhance the backup process by continuously replicating critical data to centralized servers and using tape backup systems to backup the replica rather than the production servers. Double-Take offloads the burden of periodic tape backups from multiple production servers to a dedicated backup server and makes centralized tape backup a reality, significantly reducing management cost and improving reliability. Regardless of a file's state on the source, on the target every file is closed and available for consistent backup at any point in time.



In addition to enhancing the backup process, Double-Take can take the process a step further and eliminate the often lengthy time required to restore from a tape backup by removing the tape restore process all together. Double-Take's failover capabilities allow a target machine to stand in for a source in the event of a failure, using the already online disk-based replica and removing the need to restore from tape.

Double-Take's flexibility allows many different methodologies to be implemented to suit your unique environment and needs. This document covers various backup methods and alternatives with different Double-Take configurations. It is intended for network administrators with experience installing, configuring, and maintaining network applications including Double-Take.

---

# Incremental Backups

If you are performing incremental or differential backups on your target machine, you need to make sure that your backup software is using an appropriate flag to identify what files have been updated since the last backup. For example, Double-Take does not replicate the archive bit from the source. The suggested method for incremental or differential backups is to use the last modified date on the file versus the date of the last backup.

## Registry Settings and System Files

Double-Take should not be used to replicate registry settings or system files, although Double-Take can be used to replicate file-based backups of this information. For example, you could use NTBackup on Windows 2000 to create a file containing a duplicate copy of the System State, which includes the registry. Double-Take can then replicate this file to the target. On Windows NT 4.0, you can use the Regback tool from the Windows NT Resource Kit. This tool also creates a file, containing the registry information, which can be replicated to the target. See your Windows documentation for complete details on using these commands and tools.

## Backing Up Deleted Files

A feature of Double-Take, called ignore delete operations, allows you to keep files on the target machine even if they are deleted on the source. When a file is deleted on the source, that delete operation is ignored on the target. (All edits to files on the source are still replicated to the target; only deletions of whole files are ignored.) By default, this option is disabled. Enabling it would give you the opportunity to make backups of these files in the event they are needed in the future, after they have been deleted from the source. For instructions on enabling this option, see the Double-Take *User's Guide*.

---

**NOTE:** If a file is deleted using the Windows Explorer or My Computer, from the file system perspective, the file is not actually deleted but is moved to the Windows Recycle Bin. Because Double-Take sees this as a move to outside of the replication set and not as a delete, the file will still be deleted from the target even if you have ignore delete operations enabled.

If delete operations are ignored long enough, the potential exists for the target to run out of space. In that case, you can manually delete files from the target to free space for further replication or use the automatic move or delete capabilities of Double-Take orphan files feature. For complete details on orphans files, see Double-Take's *User's Guide*.

---

---

# Applications with Interdependent Files

Backups occur sequentially from the first file to the last file. Therefore, when you are using applications that have interdependent files, such as a database application whose database and log files must be synchronized, Double-Take cannot be actively updating files on the target while the backup is running or there becomes an opportunity for interdependent files to become mismatched, causing a corrupt application on the backup.

For example, suppose the following scenario occurs on a target machine that contains a replica of a database:

1. The backup process which is currently running sequentially through the files, reaches the database log file and starts writing the file to tape. At the same time, Double-Take receives additional updates to the database. The database file is updated but since the log file is in use by the backup, the log operation is placed on the Double-Take queue on the target.
2. When the log file is finished being backed up, the backup process continues with the next file, which is not necessarily the database that corresponds with that log file.
3. Since the log file is no longer in use, Double-Take applies the log operation from the Double-Take queue.
4. Eventually, the backup process reaches the database file and writes it to tape.

At this point, the database file on the tape backup contains an extra update that the log file does not. The two files do not correspond and the database on the tape backup will not be time consistent.

The remainder of this section covers various backup methods and procedures, if applicable, for dealing with applications with interdependent files.

---

## Pausing Double-Take

Pausing and resuming the transmission of data from the source can be manually initiated through the Management Console, Text Client or you can use the DTCL commands to script it. For complete details on using the Management Console or Text Client, see the *Double-Take User's Guide*. The following instructions contain step-by-step procedures for incorporating the DTCL commands used in the Text Client into a DTCL script which will allow you to automate the pause and resume process from a batch file.

1. Prior to the backup process starting, you will want to pause transmission of operations from the source. To do this, create a batch file called `prebackup.txt` using the sample file below. Save this file to the location where Double-Take is installed.

### PREBACKUP.TXT

```
# Sample script to pause transmission of operations from the source prior to starting a #
# backup #
# Substitute the name of your source and target machines, username, password, and domain #
# in the variable definitions below. #
$TheSource = "name";
$TheTarget = "name";
$User = "username";
$Pass = "password";
$TheDomain = "domain";

login $TheSource $User $Pass $TheDomain;
login $TheTarget $User $Pass $TheDomain;
source $TheSource;
transmission pause $TheTarget;
```

---

**NOTE:** Because the following files use the `login` command, which displays password of the user ID specified, you may want to use the network administrator account. You can create a user account specifically for this purpose, add it to the Double-Take Admin security group, and grant it the minimal access necessary to complete this task.

---

2. Create a batch file to run this script using the sample batch file below.

### PREBACKUP.BAT

```
rem Sample batch file to run the prebackup.txt script

cd c:\program files\DoubleTake
cmd /c DTCL -f "c:\program files\DoubleTake\prebackup.txt"
```

- 
3. After the backup process is complete, you will want to resume transmission of operations from the target. To do this, create a batch file called `postbackup.txt` using the sample file below. Save this file to the location where Double-Take is installed.

#### POSTBACKUP.TXT

```
# Sample script to resume transmission of operations from the source after the backup is #
# complete #

# Substitute the name of your source and target machines, username, password, and domain #
# in the variable definitions below. #
$TheSource = "name";
$TheTarget = "name";
$User = "username";
$Pass = "password";
$TheDomain = "domain";

login $TheSource $User $Pass $TheDomain;
login $TheTarget $User $Pass $TheDomain;
source $TheSource;
transmission resume $TheTarget;
```

4. Create a batch file to run this script using the sample batch file below.

#### POSTBACKUP.BAT

```
rem Sample batch file to run the postbackup.txt script

cd c:\program files\DoubleTake
cmd /c DTCL -f "c:\program files\DoubleTake\postbackup.txt"
```

5. Run `prebackup.bat` before starting your backup and run `postbackup.bat` after the backup is complete.

---

**NOTE:** You can also incorporate these two scripts into an automated script that runs through your backup software.

Depending on the length of time required to complete your backup, Double-Take may not be able to queue all of the replication data. If the queue is filled, Double-Take will automatically disconnect the connections and attempt to reconnect them. This is called an auto-disconnect. If you are experiencing frequent auto-disconnects because the queues are filled while the backup is processing, you can

- ◆ Increase the size of your Double-Take pagefile
- ◆ Disable auto-reconnect and reconnect manually or in a post-backup DTCL script (It may also be desirable to script a DTCL disconnect command in a pre-backup script and reconnect in the post-backup script.)

See the Double-Take *User's Guide* for details on auto-disconnect, auto-reconnect, and increasing the pagefile.

---

---

## One-to-many replication

Another alternative to the issue with Double-Take updating files on the target while a backup is being performed is to use multiple target machines. This can be accomplished with Double-Take's one-to-many capabilities. In this configuration, you have one source machine sending data to multiple target machines. One Double-Take source/target connection can be paused or stopped (disconnected) to perform the backup, while the other Double-Take connection, from the same source to a different target, continues processing. (If replication is stopped or the connection is disconnected, restarting replication or reconnecting would require a block checksum remirror to be performed.) This configuration could separate your high availability and backup processing to two different machines. And additional machines beyond two can be added for redundant backups.

## Disconnecting Double-Take for a backup

Another alternative to the issue with files being updated on target while the backup is running, is to disconnect the Double-Take connection before the backup starts and reconnect it when the backup is finished. You can automate these processes by using Double-Take's DTCL commands to script it.

### Pre-Backup.dtcl

```
# Sample script to be run prior to initiating the backup process #
# Substitute the name of your source and target machines, username, password, domain, #
# and replication set name in the variable definitions below. (The replication set must #
# already exist.) #
$TheSource = "source_name";
$TheTarget = "target_name";
$User = "username";
$Pass = "password";
$TheDomain = "domain";
$TheRepSet = "repset_name";

login $TheSource $User $Pass $TheDomain;
source $TheSource;

# The following commands determine the connection ID from the post-backup connection and #
# then disconnects it. #
$PostConID = conid $TheRepSet to $TheTarget map exact;
disconnect $PostConID;
```

---

## Post-Backup.dtcl

```
# Sample script to be run after the backup process is complete. #
# Substitute the name of your source and target machines, username, password, domain, #
# and replication set name in the variable definitions below. (The replication set must #
# already exist.) #
$TheSource = "source_name";
$TheTarget = "target_name";
$User = "username";
$Pass = "password";
$TheDomain = "domain";
$TheRepSet = "repset_name";

login $TheSource $User $Pass $TheDomain;
login $TheTarget $User $Pass $TheDomain;
source $TheSource;

# The following line connects the replication set #
$PostConID = connect $TheRepSet to $TheTarget map exact nomirror;

# The following line starts a block checksum difference mirror. #
mirror start $PostConID different, checksum;
```

These two scripts can be incorporated into your backup process to remove administrator interaction.

---

## Rotating replicas

One disadvantage of the previous alternative, automatically disconnecting Double-Take during a backup, is that your data is not protected during the backup process. To compensate, you can take this process a step further and create multiple replicas, which are rotated, to ensure complete coverage. This can be achieved by using Double-Take's ability to connect the same replication set to multiple targets or target locations. This would grant you the time and availability of idle files on the inactive replica to perform a backup.

For example, you might have two replicas in which one is active from midnight to noon and the other is active from noon to midnight. You can start and stop these replicas using Double-Take's DTCL commands to script an automated process. The two files below are an example of two DTCL scripts which could be used to replicate data to two different locations on the same target.

### Noon.dtcl

```
# Sample script to be run at noon that stops one connection and starts a second connection.#
# Substitute the name of your source and target machines, username, password, domain, #
# and replication set name in the variable definitions below. (The replication set must #
# already exist.) #
$TheSource = "source_name";
$TheTarget = "target_name";
$User = "username";
$Pass = "password";
$TheDomain = "domain";
$TheRepSet = "repset_name";

login $TheSource $User $Pass $TheDomain;
login $TheTarget $User $Pass $TheDomain;
source $TheSource;

# The following commands determine the connection ID from the midnight connection and #
# then disconnects it. #
$MidConID = conid $TheRepSet to $TheTarget map base c:\midnight_mirror;
disconnect $MidConID;

# The following line connects the replication set and replicates the files to #
# c:\noon_mirror\source_volume_name #
$NoonConID = connect $TheRepSet to $TheTarget map base c:\noon_mirror, nomirror;

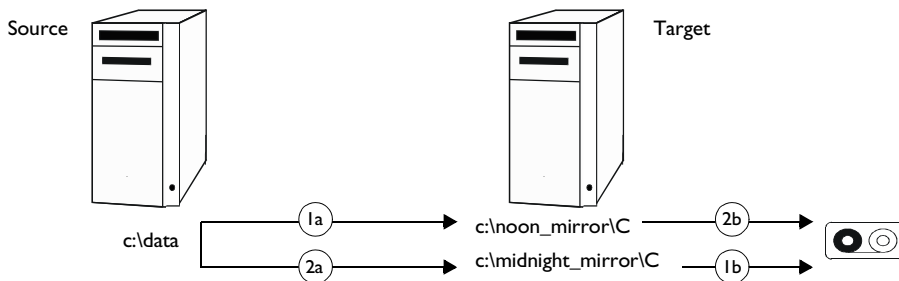
# The following line starts a block checksum difference mirror. #
mirror start $NoonConID different, checksum;
```

---

## Midnight.dtcl

```
# Sample script to be run at midnight that stops one connection and starts #
# a second connection. #
#
# Substitute the name of your source and target machines, username, password, domain, #
# and replication set name in the variable definitions below. (The replication set must #
# already exist.) #
#
$TheSource = "source_name";
$TheTarget = "target_name";
$User = "username";
$Pass = "password";
$TheDomain = "domain";
$TheRepSet = "repset_name";
#
login $TheSource $User $Pass $TheDomain;
login $TheTarget $User $Pass $TheDomain;
source $TheSource;
#
# The following commands determine the connection ID from the noon connection and #
# then disconnects it. #
$NoonConID = conid $TheRepSet to $TheTarget map base c:\noon_mirror;
disconnect $NoonConID;
#
# The following line connects the replication set and replicates the files to #
# c:\midnight_mirror\source_volume_name #
$MidnightConID = connect $TheRepSet to $TheTarget map base c:\midnight_mirror nomirror;
#
# The following line starts a block checksum difference mirror. #
mirror start $MidnightConID different, checksum;
```

These two scripts would have to be run at noon and midnight, respectively.



- 1a. At noon, the noon.dtcl script disconnects the midnight connection and starts a difference mirror and replication to the noon directory.
- 1b. At the same time, the tape backup process begins from the midnight directory.
- 2a. At midnight, the midnight.dtcl script disconnects the noon connection and starts a difference mirror and replication to the midnight directory.
- 2b. At the same time, the tape backup process begins from the noon directory.

You could expand the automation even further by using the Windows scheduler service to run the scripts. Or you could use a combination of Windows AT commands and Double-Take DTCL commands to actually connect and disconnect the replication sets at different times. These methods would also allow multiple replicas to be stored on the different targets or on the same target server but in different directories or on different drives.