PROTECTING AND ENHANCING SQL SERVER WITH DOUBLE-TAKE AVAILABILITY

Whitepaper
Double-Take Software, Inc.
Published: October, 2009

With its ease of use, scalability, reliability and broad industry support, Microsoft® SQL Server® is the solution of choice for many customers. While SQL Server 2008 offers options for improving database protection and recovery, none of them is intended to provide a complete disaster recovery solution. Today, regulatory mandates and service level agreements drive many of these requirements beyond what native SQL Server data protection options can provide. Protecting the data assets stored by SQL is the first step to providing protection and the ability to recover quickly. The second step, commonly overlooked in the initial phases of a recovery plan, is protecting the application itself.

The same real-time replication capabilities that Double-Take® Availability provides for disaster recovery and high availability can also be used in other ways. For example, many IT organizations also use Double-Take Availability to reduce or eliminate the impact of scheduled and ad-hoc reporting on their production databases when using Microsoft SQL Reporting Services. Offloading the reporting process to a secondary server can greatly increase the production server's performance and scalability. This whitepaper discusses strategies for protecting critical data stored on SQL Server and outlines strategies for enhancing Microsoft SQL Reporting Services using Double-Take Availability and a secondary, real-time copy of production SQL data sources. It also discusses the benefit of using non-production databases as data sources for Microsoft SQL Reporting Service and how using Double-Take Availability and real-time replication can help with a Microsoft SQL Reporting Services deployment.

Enhancing Native SQL Server Protection

Native SQL Server protection was never intended to be a comprehensive disaster recovery solution. The table below offers a comparison of native SQL protection and Double-Take Availability protection features.

SQL Server	Double-Take Availability
SQL Database Mirroring does not protect the entire instance. Metadata (logins, security, schema, etc.) are not included in the mirror, only data.	Automatically captures and replicates all database changes. The target server is ready without additional configuration.
SQL Agent jobs must be manually enabled upon failover with Database Mirroring.	Recovers the entire SQL Server instance - preserving the SQL Agent Job configurations.
Lacks enterprise features such as data compression, bandwidth throttling and application-independent protection, which limits the flexibility and adaptability of Database Mirroring.	Designed to operate in any environment, with features such as Intelligent Data Compression, Flexible Bandwidth Throttling, Task Command Processing, and Categorized Server Groups.

Automatic database failover requires synchronous mirroring which can have a negative impact on applications.	Uses asynchronous replication to prevent performance degradation of applications. Failover can be automatic or manual for any of the replicated servers - with no restrictions or special requirements.
Automatic failover requires an additional SQL Server instance on a separate server.	Does not require extra servers to monitor and initiate failover.
The principal and mirror servers must be in the same or trusted Windows® domains	Supports servers in mixed domains, removing restrictions for target server maintenance.
Log Shipping does NOT protect the entire SQL Server instance. Metadata (logins, security, schema, etc.) are not included in the backup, only data.	Automatically captures and replicates all database information and changes. The target server is ready to process client requests without additional configuration.
Log Shipping requires a manual, non-integrated process to copy the transaction log files to a secondary server. This is a time consuming exercise left to the DBA to build and maintain.	Captures and replicates all database changes and applies them to the target server in the exact order they were made.
L**og Shipping does not provide high availability** to protect against costly downtime. It does not detect failures or initiate any type of failover.	Failover can be automatic or manual. Recovery is within seconds, often before users realize a failure has occurred.
Log Shipping is not real-time. Data changes are copied on a periodic basis, resulting in large amounts of data at risk.	Captures and replicates changes in real-time, as they are made, to ensure the latest updates are available on the target in the event of a failure of the primary server.
Log Shipping can flood the network. It stores changes locally until the scheduled time to copy them to the secondary server, resulting in a negative impact to the production network as large amounts of data are transmitted in bulk.	Performs real-time replication, sending changes as they occur for optimal efficiency in the use of system and network resources.
Failover Clustering (MSCS) Support Distance limitations inherent to MSCS prevent it from protecting against regional or even some metropolitan failures, rendering it insufficient as a disaster recovery solution.	Does not have any distance limitations, allowing for protection against local, regional, metropolitan, and any other type of failure.
Failover Clustering requires Enterprise or DatacenterEditions of Windows, significantly increasing the cost of the solution.	All Double-Take Availability functions are certified to run under all editions of SQL Server and Windows 2000 and 2003.
MSCS requires certified hardware	Supports any hardware running Windows and can replicate data and provide failover between different types of Windows severs.

MSCS requires certified hardware	Supports any hardware running Windows and can replicate data and provide failover between different types of Windows severs.
Transactional Replication Microsoft documentation makes it clear that Transactional Replication was never intended to be a high-availability or disaster recovery solution. It does not provide the necessary features to perform these functions adequately.	Designed to be a high-availability disaster recovery solution for Microsoft applications.
SQL-native Transactional Replication does NOT protect the entire SQL Server instance. Metadata are not included in the copy process, only data.	Automatically captures and replicates all database information and changes. The target server is ready to accept and process client requests at any time without any additional configuration.
Lack of failover capabilities prevents SQL-native Transactional Replication from being used as a high availability solution. The manual process required would result in substantial downtime of the database and the applications it supports and increase the burden on the administrative staff.	Failover can be automatic or manual.
Lack of important features limits the flexibility and adaptability of SQL-native Transactional Replication to easily conform to the individual needs of businesses.	Designed for enterprise-class operation in any environment, with features that allow it to be configured for optimal performance in most any environment.
Complex architecture requires multiple servers acting in different roles, introducing points of failure into the solution.	Double-Take Availability replication servers act independently of each other and do not have a single point of failure.

How does Double-Take Availability replicate SQL databases?

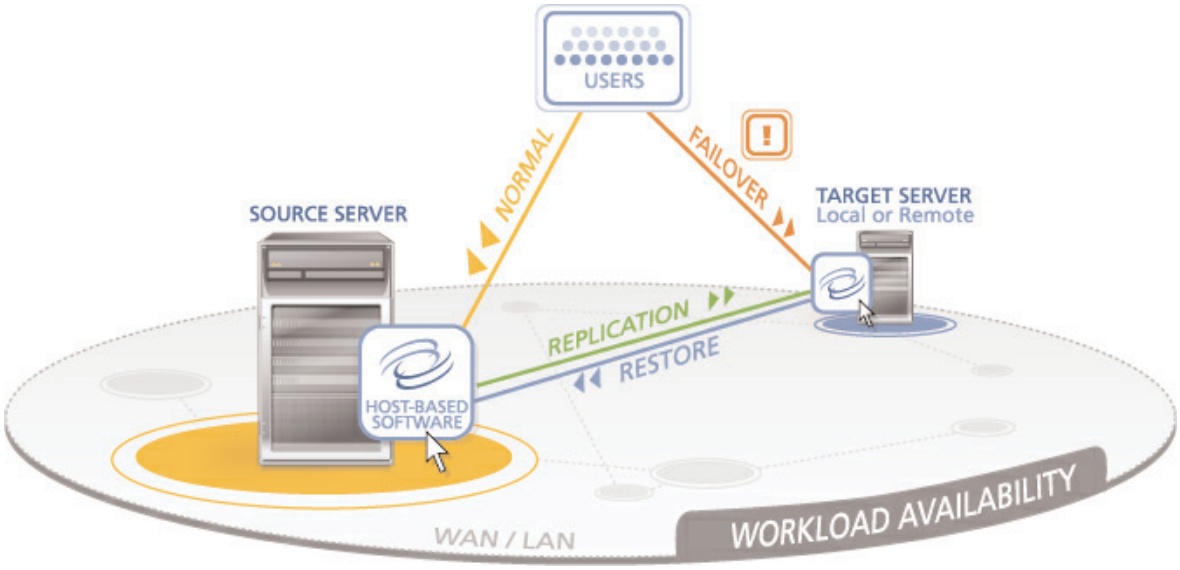
Double-Take Availability protects data stored on Windows file servers and application data used by business-critical applications like Microsoft SQL Server and Microsoft Exchange Server. It combines continuous real time replication and automatic failover capabilities for disaster recovery, high availability, and centralized backup of applications. The real-time protection of Double-Take Availability provides a Recovery Point Objective (RPO) usually measured in seconds. Double-Take Availability is Microsoft Windows 2000 and 2003 certified at all levels, one of the few replication products to have achieved this level of certification. It delivers protection that is better or comparable to many hardware based solutions, but costs tens of thousands less.

How does Double-Take Availability replicate?

Real-time protection - Replicates continuously at the byte level over any shared or private IP network, ensuring that changed data is protected and can be quickly restored at all times Application agnostic - Works with existing hardware to protect applications such as Microsoft Exchange, Microsoft SQL Server, and SharePoint - any application that runs on Windows Server.

Easily installed and maintained - Allows companies of any size looking for data protection solutions to install and maintain Double-Take Availability.

Cost effective - Provides strong data protection at low cost with an accelerated return on investment - paying for itself usually within months.



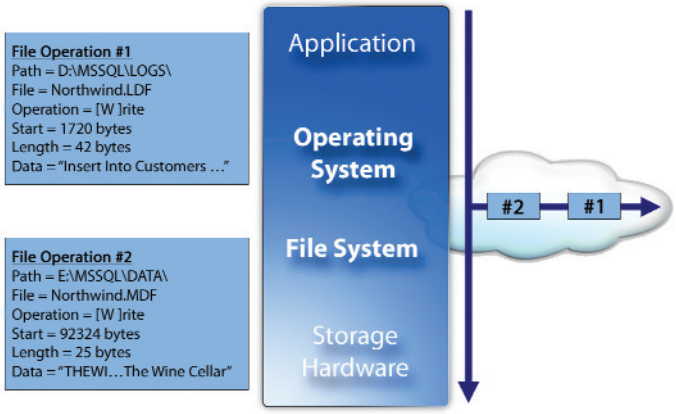
Double-Take Software has been protecting SQL Servers and replicating SQL data since SQL Server 6.0 was released and Windows NT Server 4.0 began shipping. Double-Take Availability replicates changes to files at the byte-level from any Windows Server to any other Windows Server across any IP-based network. It installs on each server and uses a file system filter driver to determine what changes are occurring to files and then replicates those file operations to another server and applies them to a secondary copy of the data. All changes are sent and applied in the exact order that they occurred on the production system, guaranteeing a crash-consistent copy of the SQL data on the secondary system.

When a transaction is written to a production server, Double-Take Availability replicates that transaction over TCP/IP and writes the entire transaction on the target as one logical unit of work regardless of the physical I/O or IP packets. Double-Take Availability also writes transactions in the exact same order on the target server as they occurred on the source. These methods ensure that a properly implemented environment will never result in a SQL Server suspect database or torn pages.

Implementing Double-Take Availability on production SQL Servers is seamless and does not require any modifications to existing databases or applications. Therefore, keyed tables are not required to provide unique rows nor does it rely on additional columns. These two stipulations are a major drawback to implementing other SQL-based replication solutions. Double-Take Availability also does not require the production databases to be off-line to provide real-time byte-level replication. This allows users to continue working normally while data replicates to another server - regardless of the distance.

How Double-Take Availability Ensures Data Integrity

All applications write data to persistent storage media such as disk drive in three basic steps: Open/Create a file, Add/Update data to the file, and Close the file. Modern operating systems provide multiple software layers that provide assistance to the application write process to ensure that the file maintains a consistent state that the application can recognize the next time that it reads the file.



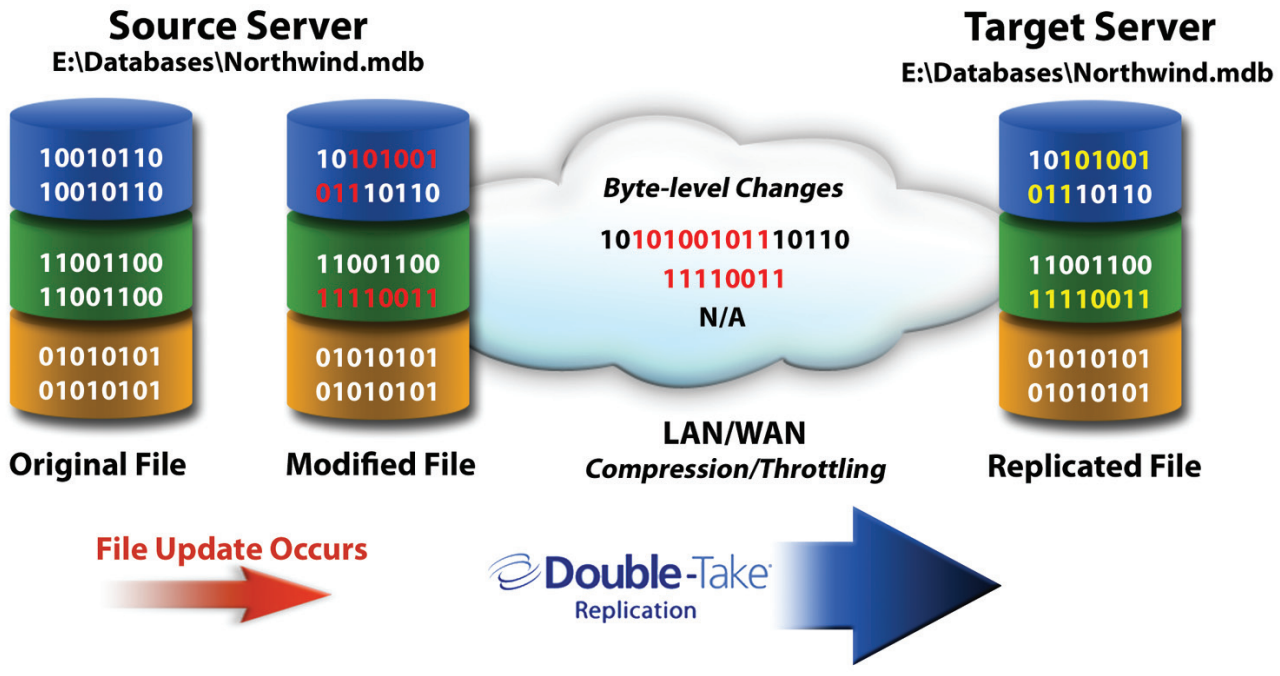
The operating system kernel provides a set of basic functions (API) that an application can use to read and write data to disk. This reduces the complexity by providing a single interface rather than a unique method for each possible type of application data or underlying media. The API provides transparency, allowing applications to write to disk, tape, Flash memory devices, etc. Most desktop applications store all of their data in memory and write an entire file to disk, relying on the file system to perform journal write protection to preserve data state. However, database applications must provide their own recovery features to preserve state.

Database applications, including SQL, Oracle™, Exchange, and Notes™ typically store much more data in their files than can fit into system memory. They open files when they start, write transactions continuously, then close files only when they are shutdown. Databases ensure that all changes made to data are protected using a write method called "Write-Ahead Logging".

The Write-Ahead Logging method of writing data requires that the transaction is written to a log on a physical disk before any commitment is made to the client that the transaction was successful. The database application can replay the transaction log to update the database when it restarts, thus reducing the probability of data loss for transactions that were confirmed to the users. Once a transaction has been written to the database, it can be marked with a checkpoint, which lets the database know that all of the instructions were successfully executed and the transaction can be removed from the log to make room for more new transactions.

```
Begin Transaction

Insert into Orders(CustName, OrderID)
  values('John Doe', 4)

Insert into Items(OrderID, Product,
  Price) values(4, 'Skis', $125.00)

Insert into Items(OrderID, Product,
  Price) values(4, 'Poles', $15.00)

Commit
```

Double-Take® Availability from Double-Take Software integrates with the Windows kernel I/O sub-system which provides a serialized view of all I/O operations before they are passed to the file system layer. This is a standard integration point where applications such as anti-virus and open file agents integrate and provides a fully consistent transactional view of all system writes.



How Double-Take Availability Failover Works

So far, this paper has addressed how applications actually write data and validated the Double-Take Availability approach to protecting it. The result is a disaster recovery or centralized backup solution for any transaction-based database, without the need for expensive application-specific agents.

The last piece of the puzzle is to provide near-immediate availability of the database from the target server for high availability, via the failover technology within Double-Take Availability. Double-Take Availability provides an alternative method for failover in situations where the application may not be "cluster-able", where a client has chosen to not implement clusters or the hardware may not support clustering. In addition, because deploying a cluster cannot be done "on top" of an existing server, many customers prefer the less obtrusive Double-Take Availability approach of simply configuring a second server, the replication target, to fail over for the production server during an outage.

To do this, Double-Take Availability provides up to five actions after determining that the production server has failed - including two script capabilities and three optional failover actions. The failover actions include assuming the production server's IP address, machine name, and UNC file shares. For local file servers, this is completely automated and results in highly available file servers. For application servers, only one additional step would be required - the restarting of the application services, such as SQL or Oracle. Just create a PostFailOver script which starts the services and the database soft recovery system handles everything else. Because it is a simple "batch file", any other tasks that are command driven can be inserted into either script for automation as part of the fail over process. The failback process is the reverse, with pre- and post- scripts plus the removal of the assumed IP address, machine name, and UNC file shares.

1. **Pre Fail Over script** - run before actual failover. Often clients will invoke a VSS snapshot or some other cleanup effort, email the helpdesk that the process has begun, or shut down non essential services to free up CPU or Memory on the target server.
2. **IP Address(es) of Source** - allows a completely transparent failover. However, if the target server and source server are on different subnets, than this step should be disabled and additional networking will be handled.
3. **Machine Name of Source** - the target server will maintain its original identity but has the capability of assuming multiple other names.
4. **UNC file shares of Source** - this allows a transparent failover of file servers. Double-Take Availability is intelligent enough to remap actual directories, such that if the Source used D:\ but the data was replicated to the Target's F:\ - the shares will be created correctly, including all relevant permissions and ACL's. Share information is updated to the target hourly.
5. **Post Fail Over script** - run after steps 2-3-4 usually for the purpose of starting services such as SQL, Oracle, Notes or Exchange.

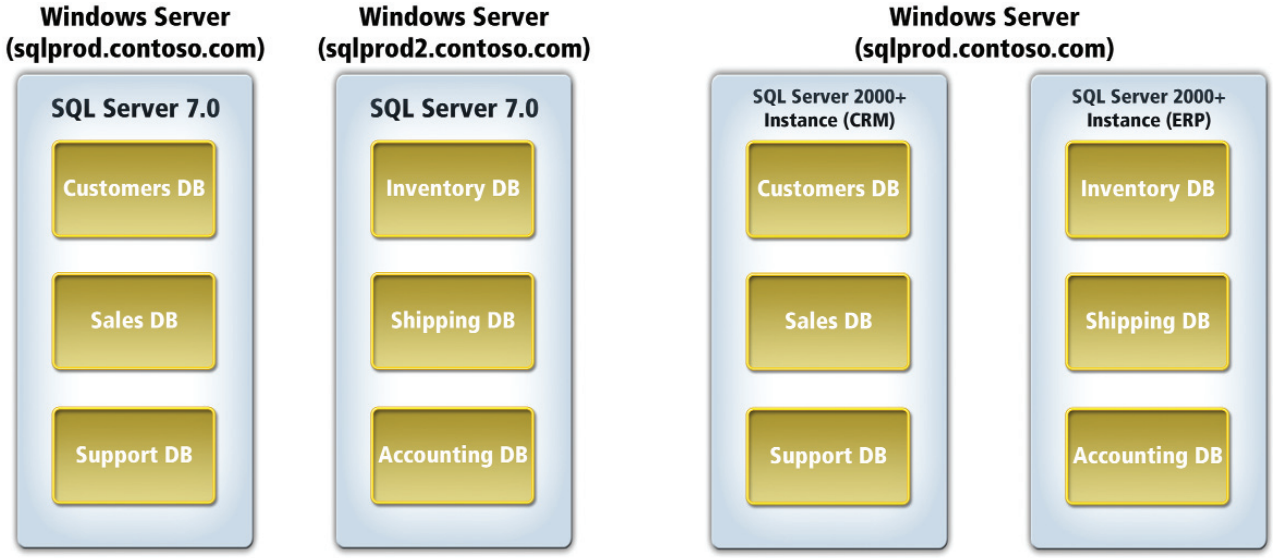
After the target has assumed the source identity, the dynamic nature of most Windows client applications will transparently and automatically reconnect to the server service, even if the client node had to re-resolve any DNS, WINS or authentication matters with the target server.

SQL Server Many-to-One Protection

In order to maintain high service levels in their disaster recovery facilities, many organizations must implement one-to-one configurations. However, some SQL Server disaster scenarios are not required to service the same level of transactions in the event of a failure. Many-to-one protection of SQL Server can be accomplished with Double-Take Availability and native features of SQL Server. There are two primary methods for providing many-to-one failover configurations: named instance protection and recovery and user protection and attachment.

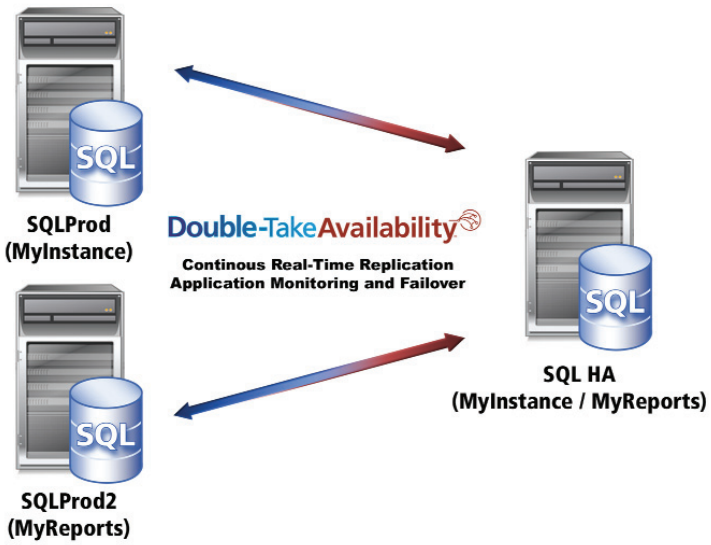
SQL Server 2000 Many-to-One Failover using Named Instances

One of the key functionalities of Microsoft SQL Server 2000, Microsoft SQL Server 2005 and Microsoft SQL Server 2008 is multiple instance support running concurrently on the same server. Each instance maintains its own set of system and user databases that are not shared between instances. Applications connect to named instances in the same way that they connect to SQL Server default (i.e. unnamed) instances except for one detail: ex. "SQLProd\CRM" and "SQLProd\ERP". The [Instance Name] portion is eliminated to connect to an unnamed instance.

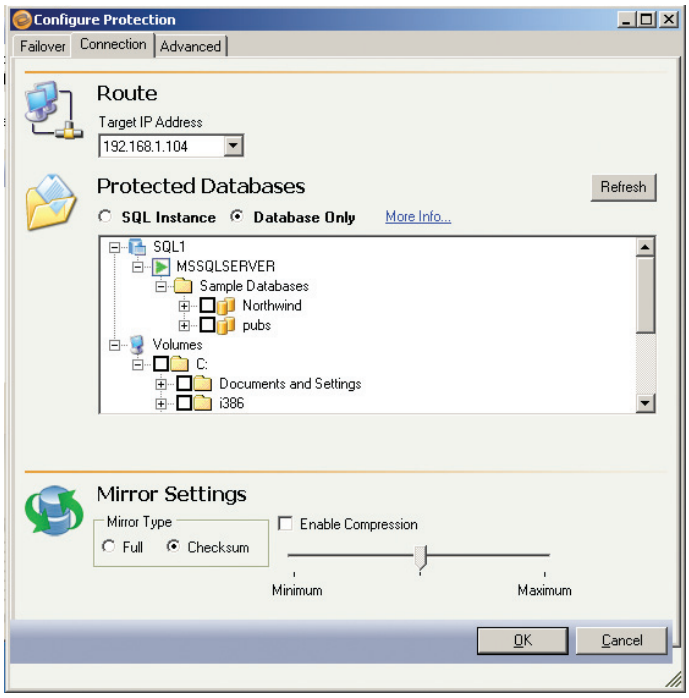


Named instances provide complete database isolation while allowing consolidation onto the same server. You can only have one default instance; however, you can have an unlimited number of named instances on a single server. Each instance must be maintained separately from the other instances on that server, including service pack updates which must be applied independently of each other. For example, if a server has three named instances, then a new service pack must be applied three times, once for each named instance. The named instance architecture provides a simple mechanism for the supporting most SQL Server 2000/2005/2008 many-to-one configurations using Double-Take Availability.

SQL Server 2000/2005/208 stores a great deal of metadata about user databases in the master, MSDB and model system databases, which store configuration information such as login and permissions for user databases. If these system databases are not recovered, then a DBA must manually reconfigure the target server. Double-Take Availability can protect the entire database instance -eliminating manual DBA re-configuration and reducing the total amount of downtime experienced during a production failure. The first configuration step is to ensure that each production SQL Server is installed as a named instance. Next, install Double-Take Availability on each source server and the target server. The target server will only have the capability of holding a single unnamed instance without requiring manual reconfiguration by a DBA. In this example we are protecting two existing production SQL Servers with named instances called SQLProd\MyInstance and SQLProd2\MyReports.



Double-Take Availability allows you to select individual files to replicate as well as directories and volumes. The data directory is located in "C:\Program Files\Microsoft SQL Server\MSSQL$[Instance Name]\Data". This directory holds all of this instance's system databases and also acts as the default location for user databases. The key files that need to be replicated are the transaction logs (.LDF files) and database files (.MDF files).



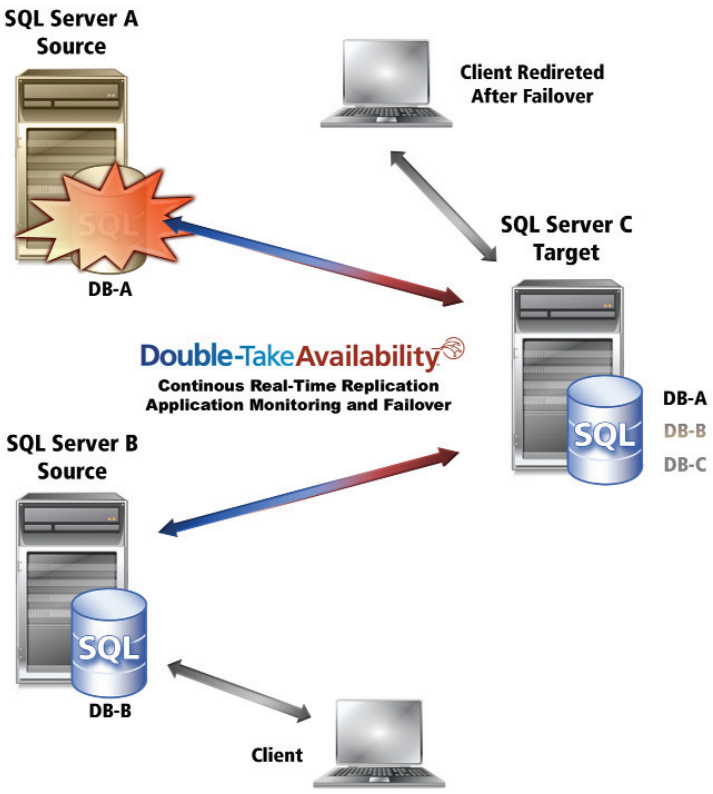
SQL Server 2000/2005/2008 should be installed on the secondary server using the same instance names as the production SQL Server instance names. Each instance should be configured to store its database and transaction log files in the same volume and path locations on the secondary server as they are located on each of the production servers. Each instance should have service packs applied separately since each instance has unique system databases and binary files. The instance's services should be set to 'manual start' so that the instances do not lock their database files exclusively while Double-Take Availability is trying to write updates to them. The default service names are "MSSQL$[Instance Name]" and "SQLAgent$[Instance Name]".

Double-Take Availability also provides fail-over functionality that will allow the secondary server to assume the network identity of the production servers and start up the SQL Server instances, allowing users to reconnect and continue working. Each production server is monitored by a heartbeat signal. If the production machine fails to respond to the configured number of heartbeats then Double-Take Availability will notify the administrator via SNMP, WMI, e-mail or the Windows Event Log that a failure was detected. Double-Take Availability can also be configured to automatically begin the failover process. If the secondary server is in an off-site location with a different subnet then the IP address should not be failed over. Double-Take Availability uses automatic DNS updates to enable these geographically disparate failover processes. Clients are then re-directed to the off-site location and reconnected to completing the failover process. For more information, see the Double-Take Availability User Guide.

SQL Server Many-to-One Failover using Database Attachment

This method of performing SQL Server many-to-one failover provides an option for implementations with very high ratios of production-to-recovery servers where customers do not want to take on the added burden of maintaining large numbers of SQL Server instances. This method uses Double-Take Availability to provide real-time data protection with crash consistent recovery of the user databases only. A database administrator must then manually configure the appropriate logins and security ACL changes to database objects since this information is stored in the system databases. System databases cannot be recovered using the database attachment recovery method. If applications use logins and security ACLs extensively, then you should strongly consider using the named instances method. Replicating and attaching production server user databases to a target server allows you to protect multiple SQL Server databases with a single machine. One or more of the production SQL Server databases can be recovered on the target server providing protection locally or spanning across countries or oceans.

On each SQL production server, install Double-Take Availability. Next, install Double-Take Availability and Microsoft SQL Server on the target systems, apply the appropriate service packs, and choose to have the SQL Server instance on the target server running or to have the instance services stopped and start up during failover. Since the Double-Take Availability Application Manager supports user database many-to-one protection, use it to configure replication/failover.



Enhancing SQL Reporting Services

First introduced for Microsoft SQL Server 2000 and now an integrated component of Microsoft SQL Server, SQL Reporting Services simplifies the process of extracting data from a wide range of sources and provides the ability to generate a variety of reports on the extracted data. It includes a reporting engine for processing and formatting the reports as well as a report designer for creating and viewing reports.SQL Reporting Services extends the capabilities of Microsoft SQL Server to provide server-based enterprise reporting. Reports can be delivered in various different formats and combined with other Microsoft products such as Microsoft SharePoint® Server and Microsoft® Office for fast data analysis and dissemination.

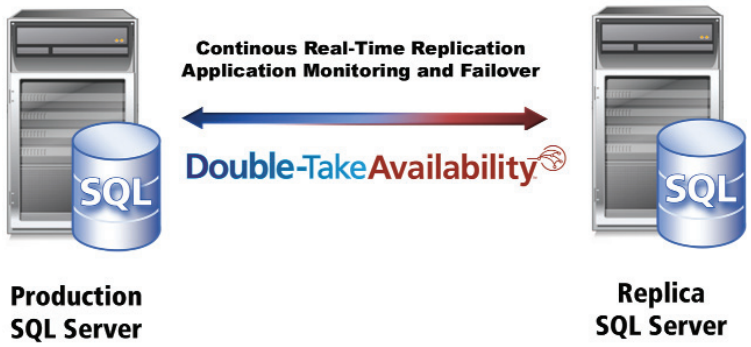
How does SQL Reporting Services interact with SQL Server?

In a typical configuration, SQL Reporting Services can be used to query "live" data from a SQL Server database or use a cached copy of the data to create the result set for a particular report. How the report is processed is determined by the report execution properties for each report and is configurable by the administrator or SQL Database Administrator. In SQL Server, there are three modes of report execution - On Demand, On Demand From Cache or From Snapshots.

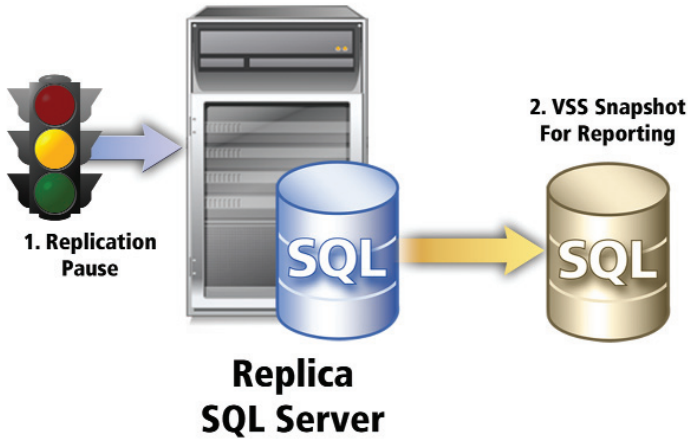
With On Demand report execution, each time a user runs a report a query is made to the report data source. This results in on-demand reports that contain the most current data. Reports executed with the On Demand From Cache option will query the data source the first time a report is requested by a user and then will cache the results of that query for a configurable period of time. Lastly, reports that are configured to run From Snapshots will use a pre-fetched copy of the results that has been retrieved at a configured schedule and saved in the SQL Reporting Services database. The last option provides the least amount of demand on the production SQL data source but potentially contains stale information.

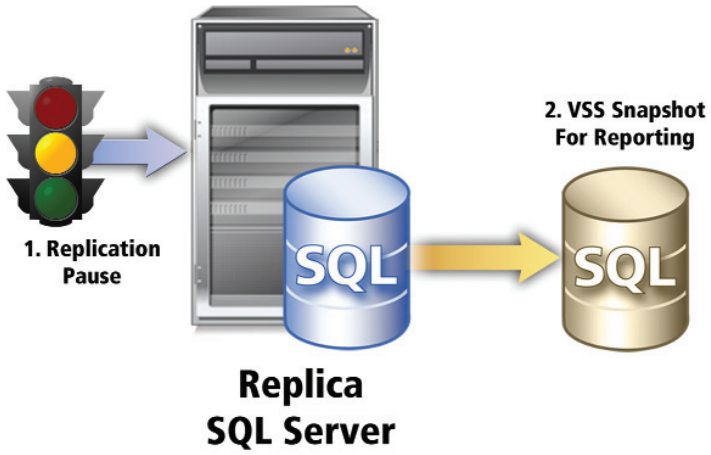
Because running a live query against production data negatively impacts the performance of the production server, cached copies or snapshots of the results are used to reduce this impact. However, the process of creating cached copies or snapshots will also impact the production server, so they need to be created during non-peak hours. By using Double-Take Availability to replicate production SQL Server data to a target server, live reports can be generated from a secondary copy of the SQL Server database being used as a Reporting Services data source. This allows reports to be created using current data without affecting the performance of the production database.

In this solution, SQL data is replicating from the production database (source server) to a database used as a Reporting Services data source (target server). A separate SQL Reporting Services server receives requests from client servers for queries and reports and then generates reports from the data on the Double-Take Availability target. The Double-Take Availability target server replica of the production SQL Server database instance is continuously in a state of change much like the production database on the Source server. However, this target replica can facilitate enhanced SQL Server functionality within the enterprise by using snapshot technology such as Windows Server's native Volume Shadow Copy Service. Prior to running a report against the target server, the connection from the source to the target is paused and a snapshot is taken of the target data. Once the SQL services are started on the target server, the Reporting Services server queries and reports against the target server data. After this query is complete, SQL services are stopped on the target server, the snapshot is restored and real-time replication resumes where it had been paused. This process could be used with scheduled Reporting Services queries at one or more times during the day so that work could be off-loaded from the production SQL server and users could experience improved performance during report generation. For on-demand queries, the target could be refreshed multiple times during the day, providing current information for users without impacting the production SQL Server.

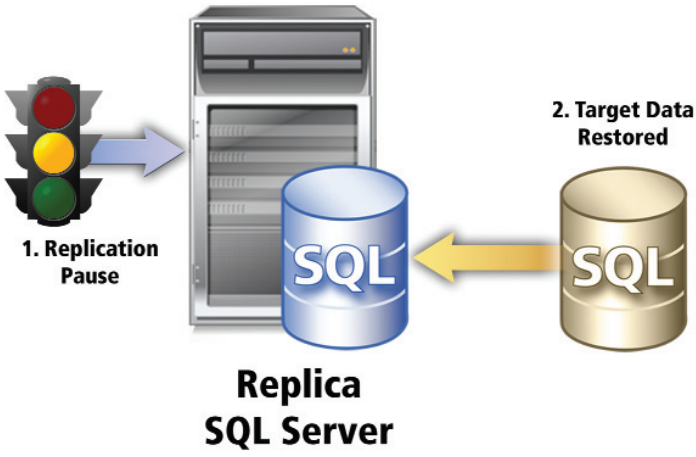


Reporting Process1. Pause replication on the target: This allows the target replica data set to reach a consistent state prior bringing the SQL Server services online. New data is still received from the production SQL Server, but is queued on the target server in the queue.2. Create a snapshot of the replica: The snapshot is for roll back of the databases to the point where replication was paused. The roll-back will occur after the reporting process is completed and the data needs to be refreshed with the new data currently stored in the queue.3. Start SQL services: SQL Server will mount the database files, bring them on-line performing its native roll-forward and roll-back functionality to flush the transaction log into the database files.4. Begin reporting: Once SQL Server is running, the reporting tools can execute their queries against the replica and build reports or allow ad-hoc reporting by end-users.





Refresh Databases1. Stop SQL services: Stopping the SQL Services forces a release of the exclusive write lock on the database and transaction log files. This will allow the previous snapshot to revert the files to their state prior to coming on-line.2. Restore snapshot: The snapshot will cause replication to resume from the point when it was paused.3. Resume Replication: Double-Take Availability will write any new data stored in the queue to the database and transaction log files. This will refresh the replica databases with all production updates that have occurred while the reporting process was taking place.



Summary

Database Management Systems (DBMS) are the hidden engines behind some of the most critical information applications - including Enterprise Resource Management (ERP), Customer Relationship Management (CRM) and accounting systems. The capabilities provided by Microsoft SQL Server and Microsoft SQL Reporting Services have the potential to revolutionize the way that companies work with information. It's important, though, for organizations to protect the availability of Microsoft SQL Server if they are to be able to have continuous access to this information. A SQL Server protection and recovery strategy supported by Double-Take Software provides up-to-the-minute data replication capabilities for all SQL data. Double-Take Software provides organizations with a solution that offers distinct recovery and protection advantages over manual Microsoft SQL built-in replication capabilities: Double-Take Availability saves more data real-time and restores that data in a much faster manner.