Technical Report

NetApp Disaster Recovery Solution for Microsoft SQL Server

Amarnath Rampratap, Sourav Chakraborthy, NetApp

Updated by John S. Parker, NetApp
January 2010 | TR-3604

NETAPP DISASTER RECOVERY SOLUTION FOR MICROSOFT SQL SERVER

This technical report describes two different scenarios that can be used to develop a disaster recovery solution for Microsoft® SQL Server® 2005 and 2008 on NetApp® storage. It is intended to be a solution-planning guide that provides SQL Server administrators and planners with the key architecture components to successfully design and deploy a disaster recovery solution for SQL Server. This TR discusses the use of NetApp SnapMirror® in conjunction with NetApp SnapManager® for SQL Server with both clustered and standalone SQL Server configurations.

TABLE OF CONTENTS

1 INTRODUCTION

Many of today's mission-critical business applications depend on Microsoft SQL Server to provide fast and reliable access to intelligent information architectures. Any downtime of these applications is disruptive to customers, employees, partners, and other stakeholders, and, as such, typically results in significant loss of revenue and productivity. Companies need to implement effective high-availability (HA) and disaster recovery (DR) solutions to minimize downtime of mission-critical applications.

This technical report delivers an overview of a disaster recovery model for Microsoft SQL Server using NetApp solutions.

1.1 PURPOSE AND SCOPE

The purpose of this technical report is to demonstrate a disaster recovery solution for Microsoft SQL Server user databases using NetApp solutions. The solution is designed to achieve multiple levels of recovery point objectives (RPOs) and recovery time objectives (RTOs). The scope of the solution is limited to the following:

1. An operational disaster recovery solution for replicating the SQL Server user databases onto a standby storage cluster and standby SQL Server instance in a secondary location
2. A fully implemented disaster recovery solution for Microsoft SQL Server based on NetApp solutions using NetApp SnapMirror for replication of the production data to the secondary (DR) site and NetApp SnapManager for SQL Server and NetApp SnapDrive® for Windows® for backup and recovery

The SnapManager for SQL Server restore functionality covered in this document is limited to an up-to-the-minute restore using NetApp volume SnapRestore® technology to recover database and transaction log volumes in the DR site.

SnapMirror replication discussed in this solution is limited to the production database only and does not cover master and MSDB databases. For more information on recovering master and MSDB databases please refer to Appendix C.

The technical report will not cover the following:

- Detailed setup of SQL Server, including Windows Failover Cluster
- Semi-synchronous SnapMirror

1.2 INTENDED AUDIENCE

This technical report is intended for information technology professionals, storage professionals, and SQL Server DBAs responsible for corporate database management infrastructure. To understand the methods and procedures mentioned in this technical report, NetApp assumes that the reader has working knowledge of the following:

- SQL Server architecture
- SQL Server storage architecture and database administration
- Service-level expertise of Microsoft SQL Server recovery options

NetApp assumes that the reader also has working knowledge of these NetApp solutions:

- Data ONTAP®
- SnapDrive for Windows
- SnapManager for SQL backup and restore procedures
- SnapMirror

2 DISASTER RECOVERY, HIGH AVAILABILITY, AND BUSINESS CONTINUITY

Business continuance is the term for the process and procedures an organization puts in place to ensure that essential functions can continue during and after a disaster. Business continuity planning seeks to prevent disruption of mission-critical services and to reinstate full functionality quickly and efficiently.

High availability is a system design protocol and associated implementation that ensure a certain degree of operational continuance during a given measurement period.

Business continuity (BC) and high availability are not specific technologies but rather integrate a variety of strategies and technologies to address all potential causes of outage that result in a resilient infrastructure that balances cost versus acceptable risk.

To appropriately architect a solution for high availability and business continuity, you should be familiar with the following terms.

AVAILABILITY

Generally, this is the degree to which a system, subsystem, service, or equipment is in an operable state for a proportion of time in a functional condition. It refers to the ability of the user community to access and use the system.

DISASTER RECOVERY

Disaster recovery is a process of regaining access to the data, hardware, and software necessary to resume critical business operations after a disaster. A disaster recovery plan should include methods or plans for copying necessary mission-critical data to a recovery site, as well as procedures to regain access to this mission-critical data after a disaster.

HIGH AVAILABILITY

HA is a system design protocol and associated implementation that ensures a certain absolute degree of operational continuity of a system, service, or equipment during a given measurement period. High-availability planning should include strategies to prevent single points of failure that could potentially disrupt the availability of mission-critical business operations.

RECOVERY POINT OBJECTIVE

The RPO describes a point in time to which data must be restored/recovered in order to be acceptable to the organization's process supported by the data.

RECOVERY TIME OBJECTIVE

The RTO is the amount of time and the service level within which service availability must be reestablished to avoid undesirable consequences associated with a break in continuity of a service/process.

SERVICE-LEVEL AGREEMENT (SLA)

An SLA is a formal, negotiated agreement between a service provider and a user (typically customers) specifying the levels of availability, serviceability, performance, and operation of a system, service, or application.

A first step in business continuity and high-availability planning is to decide which of the organization's functions are essential and must be available and operational during a crisis. Once you identify the crucial/mission-critical components, you must determine your RPOs and RTOs apportioned in terms of cost and acceptable risk.

3 SQL SERVER DISASTER RECOVERY OPTIONS

SQL Server offers many DR/HA/BC features that vary in their RPO and RTO. These options also vary in their relative complexities and resource needs. The following are the HA/BC features offered by SQL Server:

1. **Log Shipping.** Log shipping is primarily a BC solution. It involves repeated backup of a database's transactional log file and its subsequent restoration on a standby version of the database (commonly on a different SQL Server). The backup and restore are performed by scheduled jobs on the source and destination servers. Note that log shipping is a per-database BC solution. Failover is manual, after

which log shipping between the new primary and secondary servers needs to be reestablished. The connecting applications also need to be aware of both the servers involved in the log shipping pair.

2. **Database Mirroring (DBM).** Database mirroring is a full HA/BC solution in SQL Server at the database level. Source and destination SQL Server databases are known as principal and mirror, respectively. Basically, a client interacts with the principal and submits a transaction. The principal writes the requested change to the principal transaction log and automatically transfers the information describing the transaction over to the mirror, where it is written to the mirror transaction log. The mirror then sends an acknowledgement to the principal. The mirror continuously uses the remote transaction log to "replicate" changes made to the principal database to the mirror database. In the event of a disaster, the mirror is made live and connections are diverted to the new principal, that is, failover is transparent to connecting applications.

3. **Replication.** Replication may be defined as mirroring of data in database objects (for example, tables) in another database or databases. The key idea in replication is that changes made to database objects are replicated between participating SQL Servers. It should be noted that replication is performed on a per-object basis and provides fine-grained control over what data to replicate within a database. Replication may be seen as an HA/BC solution or simply BC, depending on how it is configured. However, the key function of replication is to mitigate changes made from distributed sources to shared database objects.

4. **Clustered SQL Server Installation.** When SQL Server is installed in a Windows Failover Cluster, it presents a true SQL Server HA/BC solution. Cluster failover is specific to the SQL Server service and, hence, saves the extra configuration steps of the above-mentioned HA/BC options for each database.

The following table presents a comparison of the above options.

Feature	Log Shipping	DBM	Failover Clustering	Replication
Automatic Failover	No	Yes	Yes	No
Ease of Configuration	Easy	Medium	Medium	Medium-hard
Granularity of Recovery	Database	Database	SQL Server instance	Database object
RPO	Possible data loss	No data loss	No data loss	Some data loss is possible
RTO	Time taken for database recovery	<3 seconds	20–30 seconds plus time taken to recover databases	May run into minutes
Administrative Overhead	Minimal	Checking mirror status	Maintaining cluster hardware	May get involved in cases of complex publisher-subscriber scenarios

4 DISASTER RECOVERY SOLUTION FOR SQL SERVER USER DATABASES

When architecting disaster recovery solutions for Microsoft SQL Server, it is important to review your current SLAs to derive RPOs/RTOs.

4.1 NETAPP SOLUTION COMPONENTS

SNAPDRIVE FOR WINDOWS

NetApp SnapDrive for Windows is an enterprise-class storage and data management solution for Microsoft Windows Server environments. SnapDrive enables storage and system administrators to quickly and easily manage, map, and migrate data.

NETAPP SNAPMANAGER FOR SQL SERVER (SMSQL)

NetApp SnapManager for SQL Server speeds and simplifies SQL Server data management. It empowers DBAs to utilize the capabilities of NetApp storage systems from a SQL Server-centric approach. It automates and simplifies the complex, manual, and time-consuming process associated with backup and recovery of SQL Server databases. SMSQL leverages the NetApp technology stack to create fast and space-efficient Snapshot™ copies without assistance from the storage team, leading to very high operational efficiencies.

NETAPP SNAPMIRROR

NetApp SnapMirror delivers the disaster recovery and data replication solution that today's global enterprises need. By replicating data at high speeds over LANs and WANs, SnapMirror provides the highest possible data availability and fastest recovery for mission-critical applications.

4.2 DISASTER RECOVERY SOLUTION: OBJECTIVE

The primary objective of this disaster recovery solution is to achieve the highest degree of operational continuance at the primary site, with no single point of failure. It is also to have a recovery site to which to replicate the production SQL Server databases where they can be recovered in case of a disaster. Two scenarios were tested with the above-discussed NetApp components to achieve the two different levels of RPOs/RTOs outlined below.

SCENARIO 1

To meet a near-zero-minute RPO and a five-minute RTO, the data and transaction log volumes were replicated to the DR site synchronously using NetApp SnapMirror.

SCENARIO 2

To meet a 15-minute RPO and a 20-minute RTO, SMSQL backups were scheduled and replicated to the DR site every 4 hours. SnapDrive rolling Snapshot copies of the transaction log volume were replicated to the DR site every 15 minutes using SnapMirror.

4.3 SOLUTION ARCHITECTURE

The architecture used in this model provides high availability and disaster recovery for Microsoft SQL Server operating in a primary site for production and a secondary DR site for recovery of SQL Server. In both the primary and secondary sites, SQL Server was installed on a two-node Windows Failover Cluster on a clustered NetApp storage system. Network connectivity between the NetApp storage systems in the primary and secondary sites allowed replication of data and facilitation of disaster recovery.

5 NETAPP DR SOLUTION FOR MICROSOFT SQL SERVER

Based on our experimentations, the following are the key advantages of using NetApp solutions to create a DR plan for SQL Server databases:

1. **Ease of Configuration.** The most user-friendly aspect of NetApp solutions is the ease with which the discussed DR plan can be deployed. Note that through the SMSQL GUI and a few Data ONTAP commands, a robust DR solution can be realized. This reduces administrative overhead in complex database environments.

2. **Speed and Performance.** SMSQL backups are based on volume Snapshot technology, which minimizes the duration for which the database being backed up remains frozen. This means that for very large OLTP databases, there is minimal interference with transaction processing. In addition, synchronous SnapMirror updates are also reasonably fast and allow healthy RPO and RTO figures to be achieved.

3. **Database Restoration Options.** SMSQL provides two flavors of database restoration when restoring a database using transaction log backups: point-in-time and up-to-the-minute restores. These modes provide varying degrees of recovery in between full backups in the event of sudden loss of the primary site.

4. **Simplified SQL Server System-Wide DR.** Note that by using the simplified SMSQL GUI, an administrator can opt for per-database or system-wide SQL Server DR plans. No special or extra steps need be taken for controlling the database(s) to which to impart DR capability.

The following table provides a quick snapshot of business problems and how they are addressed at the primary site to provide a resilient architecture.

Business Problem	How	Description
Single Point of Failure	Windows Failover Cluster + NetApp Storage	Windows Failover Cluster addressing server resiliency and NetApp storage cluster addressing resiliency on the storage together eliminate single points of failure on application, server hardware, and storage.
Fast Backup/Recovery	SMSQL	SMSQL automates the complex and manual backup process by creating fast and space-efficient Snapshot copies and providing faster recovery.
Disaster Recovery	SMSQL + SDW + SnapMirror	SnapMirror replicates the database and log volumes and SMSQL to provide faster backups and rapid restores.
Near-Zero-Minute RPO	SnapMirror	Scheduled full backups of the SQL Server database are replicated every four hours and the transaction log volume is replicated synchronously using SnapMirror.
Low RTO	SDW/SMSQL	Volume SnapRestore provides instantaneous restore.

6 DISASTER RECOVERY SCENARIO: SQL SERVER CASE STUDY

This section describes two scenarios, including implementation details, recovery methodologies, timelines, and failback procedures, with different RPO/RTO requirements for Microsoft SQL Server. They leverage NetApp hardware and software solutions to build a resilient infrastructure in the primary site and the DR site for recovery of SQL Server in case of a disaster.

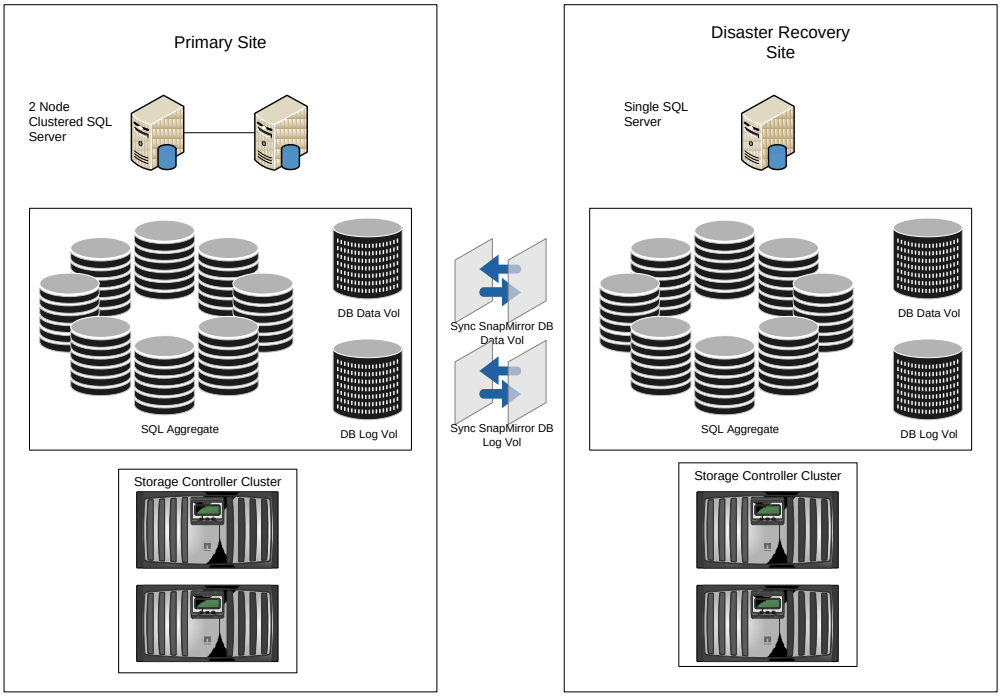
DATABASE LOAD GENERATION

All the tests discussed in the following sections used a standard SQL Server database with one MDF and one LDF file. The MDF and LDF files were placed in separate volumes. The database was put in FULL recovery mode with the “Recovery Interval” value set to default (0). The initial database size stood at 162GB. For load generation purposes, we used a custom script (please refer to Appendix A) and generated a load of 54000 tpm using OStress utility from Microsoft. The OStress utility may be obtained from http://support.microsoft.com/kb/887057. The growth rate of the database was observed to be around 6MB–7MB/sec.

6.1 SCENARIO 1: NEAR-ZERO-MINUTE RPO/FIVE-MINUTE RTO

The following section describes a disaster recovery solution for Microsoft SQL Server using NetApp solutions with the architecture discussed earlier to achieve a near-zero-minute RPO and five-minute RTO.

The following diagram shows the basic architecture used in this scenario using synchronous SnapMirror for both data and log volumes.



6.1.1 Implementation Details

As per the architecture discussed earlier, the following sections provide a detailed overview of the implementation of this scenario in the primary site and the DR site.

PRIMARY SITE

The primary objective of this setup is having the primary production site operational with no single point of failure.

- A Windows 2003 Active Directory forest was built using one domain controller.
- A two-node Windows Failover Cluster was built on two Windows Server 2003 SP2 / 2008 Enterprise Edition Servers.
- SQL Server was installed on the Windows cluster.
- SQL Server databases were hosted on a FAS6070 active-active cluster.

DR SITE

As the objective of this setup is to have an operational DR site for recovery in case of a disaster, the following were deployed:

- The minimum requirement is a standalone Windows Server 2003 SP2 / 2008 host with Microsoft SQL Server installed, but a SQL Server cluster works as well.
- A FAS6070 active-active cluster.

Storage Layout

The following layout was used in the test environment in the primary site and the DR site.

	Total Capacity	Used Capacity	LUNs
SQLDRPRI			
SQLDRDB (MDF)	**399GB**	**165GB**	**F:**
SQLDRDBLOGS (LDF)	**149GB**	**2GB**	**G:**

DATA REPLICATION

To achieve aggressive RPO/RTO targets for Microsoft SQL Server, synchronous SnapMirror updates of the transaction log and data volumes were performed.

SNAPMIRROR SOFTWARE OPERATION

SnapMirror has two distinct phases: initialization and incremental update. The initialization phase consists of a level-0 replication event in which a Snapshot copy is created on the source volume and is copied in its entirety to the target volume. The level-0 event serves to initialize or seed the mirror volume, since it contains every data block in the source volume at the time the Snapshot copy was made. The amount of time required to replicate the entire source volume to the target volume depends on many factors, including the network connection. For example, a 250GB mirror initialization could take 7.5 hours over a 100-BaseT full duplex link and may take 1.5 hours over a gigabit link.

After the mirror initialization is complete, the target storage controller examines its /etc/snapmirror.conf file every minute for any scheduled updates. This allows the mirror's configuration to be modified without disrupting the mirror. When an incremental update schedule time is due, a new Snapshot copy is created and compared with the previous Snapshot copy. The changed data blocks and the block map file are sent to the mirror target. In contrast to the level-0 initialization, the data mirrored is typically much smaller.

Note: At all times the mirror target file system is in a consistent state.

SETTING UP A SNAPMIRROR RELATIONSHIP

The following section describes how to set up a SnapMirror relationship for the test environment.
Create the SnapMirror destination volumes to be the DR site.
Note: These volumes have to be equal to or greater than the size of the source volumes.

On the source storage controller console, use the `options snapmirror.access` command to specify the host names of the storage systems that are allowed to copy data directly from the source storage system. For example:

```
options snapmirror.access host=<destination_storage>
```

Restrict the volumes to allow SnapMirror to access those using the `vol restrict` command:

```
Vol restrict <volume_name>
```

INITIALIZING THE SNAPMIRROR PROCESS

From the destination storage controller console, use the `snapmirror initialize` command to create an initial seed of the source on the destination and start the mirroring process.

```
snapmirror initialize –S <src_storage_name>:<src_vol>
<dest_storage_name>:<dest_vol>
```

Note: You can use the `snapmirror status` command to check the status of the SnapMirror initialization, as shown below:

```
FAS6070-3> snapmirror status
Snapmirror is on.
Source                    Destination       State          Lag        Status
FAS6070-1:SQLLOGS   FAS6070-3:SQLLOGSSEC   Snapmirrored   -          In-sync
FAS6070-1:sqlpridb  FAS6070-3:SQLPRIDBSEC  Snapmirrored   -          In-sync
```

CONFIGURING /ETC/SNAPMIRROR.CONF FILE

The snapmirror.conf file contains information about the updates schedule for the mirrored volume. From the destination storage controller console we did the following:

To configure synchronous replication of the transaction log and data volumes:

1. Type wrfile /etc/snapmirror.conf and hit Enter.
2. Type <Source_Storage>:<Source Volume> <Target_Storage>:<Target_Volume> - sync and hit Enter.
   For example: FAS6070-1:SQLLOGS FAS6070-3:SQLLOGSSEC - sync
   FAS6070-1:sqlpridb FAS6070-3: SQLPRIDBSEC - sync

3. Hit ctrl+c to exit.

You can use rdfile /etc/snapmirror.conf to verify the entry in the snapmirror.conf file.

6.1.2 Disaster Recovery

After creating the SnapMirror relationship between the two storage controllers, we implemented the following tests:

1. We used the script listed in Appendix A to enable load generation to take place.
2. To simulate a complete site disaster, we abruptly powered down the storage controller and the servers in the primary site by shutting off the power supplies when the load was running.

6.1.3 Recovery Methodology in the Event of a Disaster

This section outlines the recovery procedures of the SQL Server database in case of a disaster. The following steps must be taken prior to the recovery at the DR site. LUN drive letters must be the same as for the primary site.

Preparation:

1. Issue a `snapmirror break` command to the target volumes.
   Example: snapmirror break <vol_name>
2. Remove any LUN mappings carried over from the primary site.
3. Map the LUNs to the recovery server.
4. Open SQL Server Management Studio and attach the database.

Verification:

5. In order to verify the consistency of the database we ran the following script on the SQL Servers in the primary site and the DR site to calculate data loss between the sites:

```
USE SQLDRDB
Select count(*) from <table_name>
```

The results in this test scenario showed the same row counts for the test table (Tab1) in both the primary and the DR site. We also verified the status of the attached database using DBCC CHECKDB, which showed no consistency errors.

Note: If in some scenarios you're not able to attach the database, refer to Appendix B for more information on recovering the database.

6.1.4 Recovery Time

After performing a disaster simulation, the following metrics were observed at the DR site for the recovery time of the Microsoft SQL Server database.

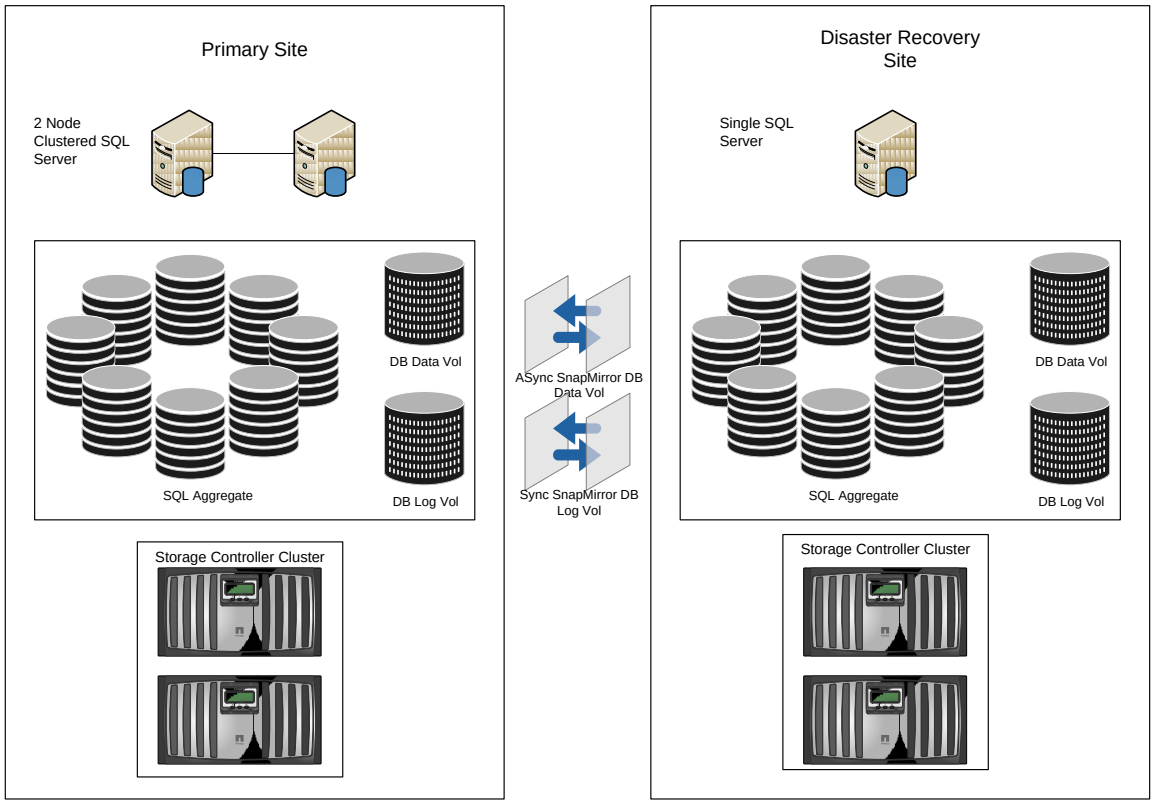
Initial Steps	Time to Completion
Break the SnapMirror for all volumes	30 seconds
Clear all LUN mappings	1 minute
Connect and mount all LUNs on SQL Server	3 minutes
Attach the database	30 seconds
Total Time	5 minutes

6.1.5 Failing Back to the Primary Site

Once the primary site is operational again, to fail back the SQL Server database you need to reinitialize the mirror using the `snapmirror initialize` command. For example:
Snapmirror initialize –S <Source_Storage>:<Source_Vol> <Destination_Storage>:<Destination_Vol>
Once the initialization is complete, place the database in single-user mode, issue a `snapmirror break` command, and recover the database as covered in section 6.1.3 above.

6.2 SCENARIO 2: 15-MINUTE RPO/20-MINUTE RTO

The following section demonstrates a disaster recovery model for SQL Server with the same architecture discussed earlier to achieve a 15-minute RPO and a 20-minute RTO.



The following diagram shows the basic architecture used in this scenario using SnapMirror to asynchronously replicate data volumes and synchronously replicate log volumes

6.2.1 Implementation Details

The implementation in this scenario is identical to that of the previous architecture except for the frequent SnapMirror updates of the transaction log backup of the SQL Server database on scheduled intervals to achieve a 15-minute RPO and a 20-minute RTO, and a SMSQL full backup of the database scheduled to run every 2 hours. The objectives remain the same.

Note: Please refer to the “Setting Up a SnapMirror Relationship” and “Initializing the SnapMirror Process” sections discussed in Scenario 1 to set up SnapMirror replication.

Scheduling SMSQL Transaction Log Backups

The following section outlines the procedures for creating and scheduling SMSQL transaction log backups.

1. Open SMSQL Manager.
2. Click on the Backup and Verification tab.
3. Select the SQL database to be backed up.
4. Select the Transaction Log Backup Radio button.
5. Click on the Schedule button.
6. Type a name for the backup job and click OK. Click No on the next two consecutive dialog boxes.
7. Open SQL Server Management Studio, expand SQL Server Agent, and expand Jobs.
8. Right-click on the transaction log backup job created by SMSQL and click on Properties.
9. Select Schedules in the left pane on the property window of the transaction log backup.
10. Click on New to schedule the job accordingly. (In this case the transaction log backups were scheduled to run every 15 minutes.)

In this test scenario it was observed that the transaction log backups were running at 45.457MB/sec, backing up 3.2GB of data in 74 seconds.

Note: SMSQL and SnapDrive do not operate concurrent operations, so it is important to make sure that the two operations do not start at the same time. If the two operations are scheduled to run at the same time, the first operation that is processed will start and the other operation will fail.

Storage Layout

The following layout was used in the test environment in the primary site and the DR site.

	Total Capacity	Used Capacity	LUNs
SQLDRPRI			
SQLDRDB (MDF)	399GB	165GB	F:\
SQLDRDBLOGS (LDF)	149GB	2GB	G:\
MASTER/TEMPDB	10GB	100MB	H:\
SNAP_INFO	179GB	33GB	I:\

DISASTER SIMULATION

Once the scheduled SMSQL backup completed we waited for 3 SnapMirror updates of the transaction log backup to complete in 15-minute intervals. At the 11th minute after the last transaction log backup Snapshot update completed we issued a `snapmirror break` command to simulate a disaster.

6.2.2 Recovery Methodology in the Event of a Disaster

This section outlines the recovery procedures of the SQL Server database in case of a disaster. The following steps must be taken prior to recovery at the DR site. LUN drive letters must be the same as those of the primary site and SMSQL must be configured ahead of time.

Preparation:

1. Issue a `snapmirror break` command to the target volumes.
   Example: snapmirror break <vol_name>
2. Remove any LUN mappings carried over from the primary site.
3. Map the LUNs to the recovery server.
4. Complete the SnapManager for SQL configuration on the DR site.
5. Perform an up-to-the-minute restore with all the transaction log backups.

The database is operational.

6.2.3 Recovery Time

After performing a disaster simulation, the following metrics were observed at the DR site for the recovery time of the Microsoft SQL Server database.

Initial Steps	Time to Completion
Break the SnapMirror for all volumes	30 seconds
Clear all LUN mappings	1 minute
Connect and mount all LUNs on SQL Server	3 minutes
SMSQL configuration	1 minute
SMSQL restore	12 minutes and 15 seconds
Total Time	**17 minutes and 45 seconds**

Note: To minimize the recovery time for the SQL Server database and restore to an operational state in the least amount of time, we performed a restore without verification and ran a **"dbcc checkdb"** from SQL Server Management Studio, which completed in 60 minutes and 7 seconds.

The total recovery time of this scenario was approximately 17 minutes and 45 seconds and the recovery point objective was 15 minutes with the SnapMirror replication of the transaction log backups to the DR site.

When planning RTOs and RPOs, it is important to know the approximate amount of transaction log data that needs to be applied and the rate at which the logs apply. In this recovery scenario, it took approximately 4 minutes and 30 seconds to apply the transaction log backups. The total restore time was 12 minutes and 15 seconds. From this data we can calculate that in this test scenario the transaction log data was applied at a rate of 2.2GB per minute.

7 CONCLUSION

To conclude our experiments, we summarize our findings in the following table.

Scenario	Impact on Database I/O Activity	SnapMirror I/O Activity	RPO	RTO
Transaction log and data volumes in synchronous SnapMirror update mode	Minimal	High	Near zero	5 minutes
Transaction log and data volumes in asynchronous SnapMirror update mode as governed by SMSQL backup schedules	High	Low	15 minutes	20 minutes

The above table gives a taste of the implications of the different SnapMirror update modes presented in this paper. Although there are many more combinations possible, for the readability of the report it was not feasible to test each one.

8 SUMMARY

Microsoft SQL Server is a mission-critical RDBMS and it can cripple operational productivity if it becomes unavailable. NetApp has proven data protection and disaster recovery tools for Microsoft SQL Server. SnapManager for SQL backup and restore capabilities and SnapDrive and SnapMirror technologies provide a solid and robust solution for protecting and recovering your SQL Server data while meeting stringent RPO and RTO objectives based on your business requirements.

9 APPENDIX A: LOAD GENERATION SCRIPTS

T-SQL Code:

```
SET NOCOUNT ON
GO

CREATE TABLE tab1 (c1 int,c2 char(100),c3 char(100),c4 char(100),c5 char(100))
GO

declare @i as bigint
set @i = 1
while @i < 10000000
begin
  insert into tab1 values(@i,'abc','def','ghi','jkl')
  set @i = @i + 1
end
GO

SET NOCOUNT OFF
GO
```

OStress Command Line:

```
C:\rml\ostress.exe -SBTC-PPE-BLADE4\SQL_2K5_DR -E -dsqldrdb -n30 -i"C:\s1.sql"
```

Here, the above T-SQL script is saved in a file named s1.sql on the C: drive.

10 APPENDIX B: RECOVERING FROM DATABASE ATTACH FAILURES ON THE DR SITE

In case of problems attaching the database at the DR site caused by checkpointing activity, please follow the procedures below. More about checkpoints may be found at the following link: http://msdn2.microsoft.com/en-us/library/ms189573.aspx.

PROPOSED WORKAROUND

The following steps need to be performed to work around checkpoint-related errors:

1. Determine that the LUN containing the data file (.MDF) is recovered and that the same drive letter that is on the primary site is assigned to it. Change the extension of the .MDF file to ".MDF.old."
2. Create a new LUN and assign it the same drive letter as the LUN that contained the TLOG file (.LDF) on the primary site.
3. Create a new database with the same name as the one on the primary site. This new database does not have to be the same size as the one on the primary site; however, it must contain the same number of data and log files (with the same file names) in the same exact drives as the original database.
4. Stop the SQL Server instance that you are working on.
5. Change the extension of the .MDF file of the newly created database to ".MDF.new" and the extension of the ".MDF.old" file to ".MDF."
6. Restart the SQL Server instance.

Note that immediately the new database comes up in Suspect mode and complains of SQL Server Error 5173.

7. Run the following command from a new query window in the SQL Server Management Studio:

```
ALTER DATABASE <name of the database being recovered> SET EMERGENCY.
Eg.) ALTER DATABASE sqldrdb SET EMERGENCY.
```

This command will bring the database into Emergency mode.

8. Detach the above database.
9. Run the following command from a new query window in the SQL Server Management Studio:

```
CREATE DATABASE <name of the database being recovered>
ON (FILENAME = '<Complete path to the .MDF file being used>')
FOR ATTACH_REBUILD_LOG
GO
Eg.)
CREATE DATABASE SQLDRDB
ON (FILENAME = 'F:\Microsoft SQL Server\MSSQL.1\MSSQL\Data\sqldrdb.mdf')
FOR ATTACH_REBUILD_LOG
GO
```

The above command will create a new TLOG file on the same drive as the one that has the .MDF file above.

10. Run DBCC CHECKDB on the present database to confirm whether or not there is any corruption.

TESTS AND RESULTS

We tested the above workaround on the same testbed as that of Scenario 1 and were able to bring the database online on the DR site.

NOTE: The above-mentioned workaround may result in data loss subject to the recovery.

11 APPENDIX C: REFERENCES

Microsoft SQL Server
Description of Disaster Recovery Options for Microsoft SQL Server

INF: Disaster Recovery Articles for Microsoft SQL Server

Disaster Recovery

Introduction to Backup and Restore Strategies in SQL Server

You Cannot Restore System Database Backups to a Different Build of SQL Server

SnapManager for SQL
NetApp SnapManager for SQL 2.1 Installation and Administration Guide

Best Practices Guide: Microsoft SQL Server 2000/2005 and NetApp Solutions

SnapDrive for Windows
SnapDrive for Windows 5.0 Installation and Administration Guide

SnapDrive for Windows Best Practices Guide

Data ONTAP
Data ONTAP System Administration Guide

Data ONTAP Storage Management Guide

NetApp SnapMirror
SnapMirror How-to Guide

SnapMirror Best Practices Guide

SMSQL Best Practices Guide

Database Layout with Data ONTAP 7G

www.netapp.com