

# Deployments and Tests in an iSCSI SAN

SQL Server Technical Article

**Writer:** Jerome Halmans, Microsoft Corp.

**Technical Reviewers:** Eric Schott, EqualLogic, Inc.

 Kevin Farlee, Microsoft Corp.

 Darren Miller, EqualLogic, Inc.

**Published:** June 2007

**Applies To:** SQL Server 2005

Summary: iSCSI SANs offer an alternative for building Storage Area Networks. Consolidating storage in a SAN offers storage management and scaling benefits for datacenters. iSCSI support in Windows Server 2003 makes connecting servers to an iSCSI SAN easy and affordable. This paper describes the deployment and testing results of SQL Server 2005 using the Microsoft iSCSI Initiator with an EqualLogic iSCSI SAN. It helps you understand best practices and the benefits of using an iSCSI SAN with SQL Server 2005.

### Copyright

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

 2007 Microsoft Corporation. All rights reserved.

Microsoft, Windows, and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

All other trademarks are property of their respective owners.

### Table of Contents

[Introduction 2](#_Toc169804514)

[Technology Background 2](#_Toc169804515)

[Configuration Recommendations 3](#_Toc169804516)

[Hardware Considerations 3](#_Toc169804517)

[Network Considerations 3](#_Toc169804518)

[Test Environment Configuration 5](#_Toc169804519)

[Database and Host Configuration 5](#_Toc169804520)

[iSCSI SAN Storage Configuration 6](#_Toc169804521)

[Tests Performed 6](#_Toc169804522)

[SQLIO Results 7](#_Toc169804523)

[SQLIOSim Results 9](#_Toc169804524)

[Conclusion 10](#_Toc169804525)

[Appendix 11](#_Toc169804526)

[SQLIO Configuration Scripts 11](#_Toc169804527)

[SQLIOSim Utility Test Configuration 12](#_Toc169804528)

#### Introduction

Internet SCSI (iSCSI) is an industry standard developed to enable the transmission of SCSI block storage commands and data over an IP network by using the TCP/IP protocol. The new method of building a Storage Area Network (SAN) offers many advantages, including lower capital costs, more familiar infrastructure for IT administrators, and excellent integration with Microsoft® Windows Server® environments.

Many administrators have questions on both the deployment and operation of iSCSI SANs with applications such as Microsoft® SQL Server™ 2005. Administrators want to know how iSCSI behaves in typical SQL Server transaction environments. In this paper we tested two Microsoft utilities geared toward simulating specific I/O patterns and SQL Server 2005-like workloads, and report both the qualitative and quantitative results.

We decided to test SQL Server in this environment and found it to be simple to operate and comparable in performance to other storage networking implementations. The Microsoft iSCSI Initiator and EqualLogic iSCSI storage arrays were easily configured and able to sustain heavy SQL transaction loads. SQL Server functioned well and required no special changes. While overall system performance requires proper server and SAN network configuration, these are easily done with today’s server and storage technologies.

The results show that iSCSI performs well with both light and intense SQL Server 2005 transaction workloads. The results also show that the Microsoft iSCSI Initiator can be used in all SQL Server 2005 workloads with excellent results.

##### Technology Background

iSCSI is a network protocol standard that allows the use of SCSI over TCP/IP networks. Because IP networks are ubiquitous, iSCSI can be used to transmit data over LANs, WANs, or the Internet, and can enable location-independent data storage and retrieval.

While iSCSI enables a new deployment for storage networking, it still uses the SCSI command set—but changes the transport carrying data to and from the SQL Server system. Because of the TCP/IP infrastructure, iSCSI deployments tend to be less expensive than Fibre Channel networks—infrastructure costs are lower, and most administrators are already familiar with managing TCP/IP environments. In iSCSI, 1‑GB (gigabyte) or 10‑GB Ethernet are typically used for data transmission. As in all SQL Server storage environments, storage system configuration and operation are critical to overall SQL operations and performance. Whether you select Fibre Channel or gigabit Ethernet connectivity for your SAN, proper storage configuration is key to effective SQL Server performance and reliability.

Historically, larger SAN deployments relied on Fibre Channel storage networking; recently iSCSI has become available across a variety of enterprise storage arrays, offering a wide variety of choices for SAN storage. iSCSI can be deployed for any size database, and supports all normal SQL functions including transaction systems, data warehouses, cluster configurations, Multipath I/O (MPIO), Microsoft Volume Shadow Copy Service (VSS), and Microsoft Virtual Disk Service (VDS) technologies.

#### Configuration Recommendations

These tests were run by using SQL Server 2005 with Service Pack 1 (SP1) running on Windows® 2003 with SP1; the only fixes implemented were for security. We used the Microsoft iSCSI software Initiator version 2.01 and Cisco 3570G Ethernet switch. The iSCSI SAN consisted of three EqualLogic PS Series 3800XV arrays containing a total of 48 15K RPM Serial-attached SCSI (SAS) disk drives.

##### Hardware Considerations

SQL Server 2005 hardware deployment best practices must be followed for optimal operations—no changes were needed to accommodate the iSCSI environment. All tests were run using the Microsoft iSCSI software Initiator with three dedicated network interface cards (NICs) for iSCSI traffic and two additional NICs for client and crossover networks.

Another option for iSCSI deployments is iSCSI Host Bus Adapters (HBAs). HBAs can benefit SQL Server by offloading the iSCSI processing resources from the server CPU onto the adapter card. In high-performance database environments this may drastically improve performance for both server resources and iSCSI throughput.

It is important to properly configure your Windows 2003 server for optimal memory utilization—insufficient memory for SQL Server generates heavier I/O loads to the database, while sufficient memory enables more efficient performance of server and storage resources.

##### Network Considerations

It is recommended that the iSCSI SAN network be separated (logically or physically) from the data network for SQL Server workloads. This ‘best practice’ network configuration optimizes performance and reliability for both SQL Server and Windows. It is possible to isolate the iSCSI and data networks that reside on the same switch infrastructure through the use of VLANs and separate subnets. Redundant network paths from the server to the storage system via MPIO will maximize availability and performance.

In transactional environments SQL I/O size is typically 8 KB, but in decision-support implementations the more sequential I/O can be up to 256 KB. Under these circumstances, it is beneficial to use multipath I/O to increase bandwidth to and from the server to the storage array for optimal database performance and reliability. MPIO should be set for round robin (Microsoft MPIO DSM) or least queue depth (EqualLogic MPIO DSM) load balancing to allow all paths to be used.

For these tests, flow control was set to Generate & Respond on all NICs to allow hardware-based adjustments that prevent dropped packets. Setting flow control is highly recommended and helps resolve in an efficient manner any imbalance in network traffic between sending and receiving devices (another best practice). Adding receive descriptors/buffers on the server’s Ethernet drivers can also help prevent dropped packets. (See your NIC manufacturer’s recommendations for increasing the NIC buffers size, as well as enabling any offload features that may be present.) Command-line tools such as netstat can help you analyze the server’s network by displaying active TCP connections, Ethernet statistics, the IP routing table, and additional IP protocol information.

In addition, for this report jumbo frames were enabled on all NICs to help reduce the interrupt rate overhead on the SQL Server systems. This is not required for proper operation, but can provide modest benefits on the server CPU utilization.

Jumbo frames and flow control were also enabled on the Gigabit Ethernet switch (mandatory if enabled on the host NICs). Additional network considerations include minimizing switch hops and maximizing the bandwidth on the inter-switch links if present. Reducing the number of switch hops between the server(s) and the storage reduces the chances of storage traffic competing with other data traffic on congested interswitch links. To avoid bottlenecks, interswitch links should be sized properly and use stacking cables, 10-Gigabit Ethernet uplinks, or link aggregation or port trunking. If multiple switches or switch blades are used, the network cables can be connected to separate switches to provide switch protection as well as increased bandwidth.

Table 1   Configuration details

|  |  |
| --- | --- |
| **Component** | **Details** |
| **Router** | One Cisco 3570G switch |
| **iSCSI initiator** | Microsoft iSCSI Initiator version 2.01 |
| **Frame size** | Jumbo frames enabled on all connected devices. Size set to 9014 bytes |
| **Flow control** | Flow control set to Generate & Respond on all network interface cards |
| **Receive descriptors** | Set to 2048 on all network interface cards |

#### Test Environment Configuration

The configuration diagram in Figure 1 shows the iSCSI and client network setup used for these tests. The iSCSI network consisted of EqualLogic PS Series storage arrays with three network ports configured on each array and the SQL Server 2005 server with three dedicated iSCSI network ports configured. The private or client network was separated from all iSCSI traffic and used for client connections to the server system as well as outside access to other networks via a separate network.



**Figure 1   Test environment**

##### Database and Host Configuration

Configuration details are described in Table 2. SQL Server was left at all default configurations, and no tuning was done; however, the Windows Server 2003 X64 Enterprise network was tuned. While VSS is available, it was not tested for this report.

**Table 2   Database and host configuration details**

|  |  |
| --- | --- |
| **Component** | **Details** |
| **Operating system** | Microsoft Windows Server 2003 Enterprise x64 Edition |
| **System Type** | Intel x64 |
| **Database server** | Microsoft SQL Server 2005 Enterprise Edition |
| **Processor** | 8 \* 3000 MHz |
| **Total Physical Memory** | 32 GB |
| **Network interfaces** | Three Intel(R) PRO/1000 MT Dual Port Network Connection adapters dedicated for private iSCSI trafficOne 1000‑MBps interface connected to public network |

It is important to properly configure your Windows 2003 server for optimal memory utilization—insufficient memory for SQL Server generates heavier I/O loads to the database, while sufficient memory enables more efficient use of storage resources.

Volume layout should optimize the manageability of SQL Server depending on the size of your environment and your array grouping. Of course, volume sizing also depends on your particular space requirements and backup/recovery needs.

##### iSCSI SAN Storage Configuration

The tests were conducted using NTFS, and disks were aligned in advance according to EqualLogic best practices. (Disk partitions should be aligned to enable optimal SQL Server and Windows Server 2003 performance.) The storage array was configured for RAID 10, the recommended RAID level for large SQL Server transactional implementations that need optimal performance. RAID 50 is available for environments requiring good performance while maximizing storage capacity.

**Table 3   iSCSI SAN configuration**

|  |  |
| --- | --- |
| **Component** | **Details** |
| **Disk arrays** | Three EqualLogic PS 3800XV arrays  |
| **Disks** | 42 15KB RPM drives (in use, 48 total) |
| **Storage Pools** | One |
| **Volumes** |

|  |  |  |
| --- | --- | --- |
| **Name** | **RAID configuration** | **Size** |
| Testvol | Automatic | 1 terabyte |
| Logs | Automatic | 100 GB |

 |

#### Tests Performed

Two key tests were run to validate the SQL Server environment. The SQLIO Disk Subsystem Benchmark Tool (SQLIO) and SQLIOSim are utilities that stress and simulate SQL Server‑like workloads. SQLIO is designed to measure the I/O capacity of a given configuration, and to verify that your I/O subsystem is functioning correctly under heavy loads. Performance numbers could be derived from System Monitor, but the tool is useful because it can be throttled. The SQLIO tests examine the following:

* How random and sequential I/O perform in an iSCSI environment
* How throughput is affected by the number of network links
* How MPIO affects performance and iSCSI traffic

The SQLIOSim utility is designed to generate exactly the same type and patterns of I/O requests to a disk subsystem as SQL Server would, and to verify the written data exactly as SQL Server would. Note that this is an accuracy and stress tool, *not* a performance measurement tool.

##### SQLIO Results

The overall SQLIO runs revealed that the system tested has the potential to move a substantial amount of data. The results show that iSCSI SAN environments can perform as well or better than Fibre Channel SAN environments for database operations and transactional workloads.

The random I/O tests reached 13,699 IOPS for read operations and 8,551 IOPS for write operations based on 8‑KB I/O sizes and 42 active physical drives. Regardless of the number of paths configured, the highest throughput was derived by increasing the queue depth to 32 or more.

The following graph shows the results of the SQLIO Random I/O tests. The values of the x-axis are difficult to read due to the number of values in the graph. For clarity, the high throughput peaks are the results of the tests throttling the queue depth to 32 outstanding I/Os per configuration. The low throughput peaks are the test results with a queue depth of four outstanding I/Os per configuration. All tests were run with 8‑KB I/O sizes.

 

**Figure 2   Random SQLIO performance**

The sequential I/O tests were run using I/O sizes of 64 KB, 128 KB, and 256 KB. In this case, the queue depth was not increased and the results reveal a much different conclusion. Based on the graph in Figure 3, it is clear that the number of configured paths had a much greater impact on throughput results. As the number of configured paths to the storage group increased, the performance and throughput increased. For optimal iSCSI SAN performance, plan for adequate network resources on the host system to accommodate high I/O type applications. As expected, the results for IOPS were directly related to the size of the I/O being run. The smaller the I/O, the more input/output operations can be performed.



Figure 3   Sequential SQLIO performance

To expand on these results and clearly show the difference between using a single I/O path and multiple I/O paths, the following graph shows the performance of the 64‑KB sequential I/O test results. These results show a range from 1700-1900 R/W IOPS at the low end with a single path, scaling up to 3600-3800 R/W IOPS at the high end with three paths connected to the target volume.



**Figure 4: Multipath I/O results**

##### SQLIOSim Results

As stated earlier, SQLIOSim accurately simulates the I/O patterns of a SQL Server environment. (SQLIOSim is an updated version of the SQLIOStress utility used in the past.) SQLIOSim was run for 48 hours to test the validity of the SQL Server 2005 environment operating in an iSCSI SAN. During the test, the SQL Server 2005 configuration ran and operated without failure or incident. The configuration parameters used for the SQLIOSim test are listed at the end of this document in the [Appendix](#_Appendix).

#### Conclusion

These tests demonstrate that iSCSI is a viable storage configuration for SQL Server and database application deployment. SQL Server performed well and behaved as expected throughout all the tests. Like all database deployments, when servers, networks, and storage are properly deployed, the results show overall synergy of the total environment.

The EqualLogic PS Series storage arrays that were tested proved that performance and scalability are easy to achieve in iSCSI SAN deployments. Database I/O and throughput are independent of the SCSI protocol transport mechanism but rely heavily on sufficient hardware and software resources to access targets or volumes across the infrastructure.

For anyone choosing to implement an iSCSI SAN, WHQL-qualified iSCSI storage arrays are listed at [Microsoft Storage Technologies – iSCSI](http://www.microsoft.com/windowsserver2003/technologies/storage/iscsi/default.mspx) on Microsoft.com.

It is important to remember that WHQL certification does not differentiate among storage arrays in terms of functionality—users should select WHQL-qualified arrays according to the performance, reliability, scalability, and the features that you require.

**Feedback**

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

* Are you rating it high because it has good examples, excellent screenshots, clear writing, or another reason?
* Are you rating it low due to poor examples, fuzzy screenshots, unclear writing?

This feedback will help us improve the quality of white papers we release. Send feedback.

#### Appendix

##### SQLIO Configuration Scripts

Sequential I/O Test Scripts

sqlio -kW -s360 -fsequential -o8 -b64 -LS -Fparam.txt

timeout /T 60

sqlio -kW -s360 -fsequential -o8 -b128 -LS -Fparam.txt

timeout /T 60

sqlio -kW -s360 -fsequential -o8 -b256 -LS -Fparam.txt

timeout /T 60

sqlio -kR -s360 -fsequential -o8 -b64 -LS -Fparam.txt

timeout /T 60

sqlio -kR -s360 -fsequential -o8 -b128 -LS -Fparam.txt

timeout /T 60

sqlio -kR -s360 -fsequential -o8 -b256 -LS -Fparam.txt

Random I/O Test Scripts

sqlio -kW -s360 -frandom –o4 –b8 -LS -Fparam.txt

timeout /T 60

sqlio -kW -s360 -frandom –o8 –b8 -LS -Fparam.txt

timeout /T 60

sqlio -kW -s360 -frandom –o16 –b8 -LS -Fparam.txt

timeout /T 60

sqlio -kW -s360 -frandom –o32 –b8 -LS -Fparam.txt

timeout /T 60

sqlio -kR -s360 -fsequential –o4 –b8 -LS -Fparam.txt

timeout /T 60

sqlio -kR -s360 -fsequential -o8 –b8 -LS -Fparam.txt

timeout /T 60

sqlio -kR -s360 -fsequential –o16 –b8 -LS -Fparam.txt

timeout /T 60

sqlio -kR -s360 -fsequential –o32 –b8 -LS -Fparam.txt

##### SQLIOSim Utility Test Configuration

SQLIOSim Configuration Settings for *sqliosim.cfg* File Parameters

|  |  |  |
| --- | --- | --- |
| **[CONFIG]** | **[RandomUser]** | **[AuditUser]** |
| ErrorFile=sqliosim.log.xml | UserCount=8 | UserCount=2 |
| CPUCount=8 | JumpToNewRegionPercentage=500  | BuffersValidated=64 |
| IOAffinity=0 | MinIOChainLength=50 | DelayAfterCycles=2 |
| MaxMemoryMB=30957 | MaxIOChainLength=100 | AuditDelay=200 |
| StopOnError=TRUE | RandomUserReadWriteRatio=9000 |  |
| TestCycles=1 | MinLogPerBuffer=64 |  |
| TestCycleDuration=172800 | MaxLogPerBuffer=8192 |  |
| CacheHitRatio=1000 | RollbackChance=100 |  |
| NoBuffering=TRUE | SleepAfter=5 |  |
| WriteThrough=TRUE | YieldPercentage=0 |  |
| MaxOutstandingIO=1000 | CPUSimulation=FALSE |  |
| TargetIODuration=20 | CPUCyclesMin=0 |  |
| AllowIOBursts=TRUE | CPUCyclesMax=0 |  |
| UseScatterGather=TRUE |  |  |
| ForceReadAhead=TRUE |  |  |
| DeleteFilesAtStartup=TRUE |  |  |
| DeleteFilesAtShutdown=FALSE |  |  |
| StampFiles=FALSE |  |  |

|  |
| --- |
| **[ReadAheadUser]** |
| UserCount=2 |
| BuffersRAMin=32 |
| BuffersRAMax=64 |
| DelayAfterCycles=2 |
| RADelay=200 |

|  |
| --- |
| **[BulkUpdateUser]** |
| UserCount=0 |
| BuffersBUMin=64 |
| BuffersBUMax=128 |
| DelayAfterCycles=2 |
| BUDelay=10 |

|  |
| --- |
| **[ShrinkUser]** |
| MinShrinkInterval=120 |
| MaxShrinkInterval=600 |
| MinExtends=1 |
| MaxExtends=20 |

|  |
| --- |
| **[File1]** |
| FileName=Q:\SqlIOSim.mdx |
| InitialSize=50000 |
| MaxSize=102400 |
| Increment=100 |
| LogFile=FALSE |
| Sparse=FALSE |

|  |
| --- |
| **[File2]** |
| FileName=L:\sqliosimlog.ldx |
| InitialSize=1000 |
| MaxSize=2000 |
| Increment=100 |
| Shrinkable=FALSE |
| LogFile=TRUE |
| Sparse=FALSE |