Comparison of Performance of Competing Database Storage Technologies: NetApp Storage Networking vs. Veritas RAID

Dan Morgan, O.C.P. M.C.S.E. | Network Appliance | TR 3105

TECHNICAL REPORT

Network Appliance, a pioneer and industry leader in data storage technology, helps organizations understand and meet complex technical challenges with advanced storage solutions and global data management strategies.

Table of Contents

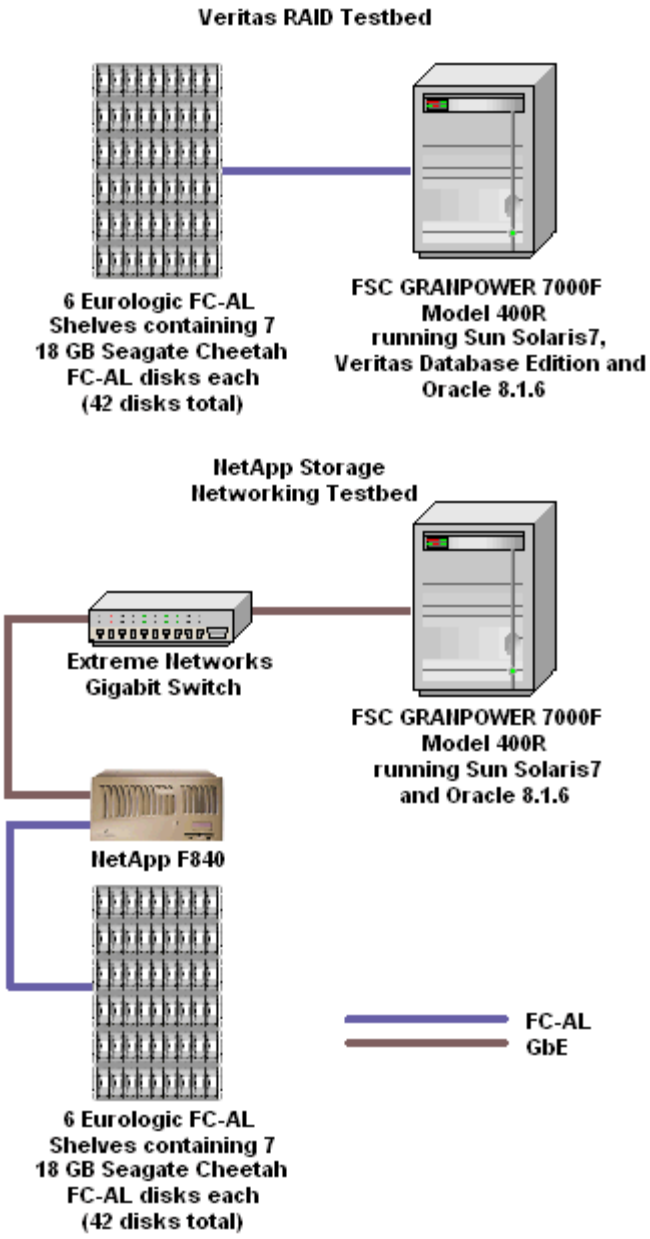
---

1. Executive Summary

This paper documents a set of performance tests on competing database storage technology. The following two technologies were the focus of this testing:

- Storage networking using Network Appliance filers
- RAID using Veritas with Quick I/O

The following network diagram contains the configuration used in these tests:



The following matrix contains the results of this testing:

Metric	NetApp F840	Veritas (RAID 0+1)	Veritas (RAID 5)
Average Response Time	.21 sec	.25 sec	.28 sec
Disk Capacity Utilized	24%	43%	21%
Disk Throughput Utilized	50%	68%	65%
Disk Access Times	13 ms	28 ms	21 ms
Disk Volume Creation	< 2 min	3.2 hr	6.5 hr
Database Creation/Load	26.95 hr	50.97 hr	61.30 hr

Transactions per Interval	28,360	24,695	24,085

The balance of this paper contains an overview of the technologies being compared, a more detailed description of the tests performed, and technical details of the configuration used in these tests.

2. Database Storage Technologies Background

2.1. JBOD

Traditionally, the database storage market was dominated by the use of JBOD (Just a Bunch Of Disks) technology. This consists of disks connected to a database server by means of a SCSI or Fibre Channel controller, without the use of RAID (Redundant Array of Inexpensive Disks). While JBOD technology is well understood, it has many disadvantages, including a lack of HA (High Availability) and difficult, expensive tuning. For this reason, the use of JBOD for storing database files is declining, and it is being replaced by RAID and storage networking.

2.2. RAID

RAID solutions address most of the deficiencies of JBOD. RAID systems balance load across a set of spindles, providing a measure of automatic load balancing. When a single disk drive in a RAID array fails, the RAID system can recover the data without any need to resort to restoring a backup. These advantages come at some cost in terms of storage efficiency and performance. The dominant forms of RAID are RAID 5 and RAID 0+1 (also known as mirrored stripping).

The market for RAID storage products is undergoing rapid growth. This growth has been driven by three factors:

1. The growth of processor speed has outstripped the growth in disk speed. This imbalance transforms traditionally CPU-bound applications to disk I/O-bound applications. To obtain an improvement in application performance, disk I/O bandwidth must be increased. The most common way to do this is by increasing the number of disks used to work on the problem.
2. Arrays of small diameter disks often have substantial cost, power, and performance advantages over larger disk drives.
3. Disk array subsystems can be made highly reliable by storing a small amount of redundant information in the array. Without this redundancy, large disk arrays have unacceptably low data reliability because of their large number of components disks. This is the reason RAID was developed.

The most common variant, RAID 5, employs distributed parity. Data is stripped over all disks so that large files can be fetched with high bandwidth. By distributing the parity, many random blocks can be written in parallel without creating a hot disk.

The other most common variant is RAID 0+1, which combines nonredundant striping with mirroring. The principal disadvantage of this technology is cost, because the disk overhead is 100%.

While RAID 5 disk arrays offer performance and reliability advantages for a wide variety of applications, they have at least one critical limitation: their throughput is penalized by a factor of four over nonredundant arrays for workloads of mostly small writes. This penalty arises because a small write request requires the following steps to be performed:

1. The old value of the user's targeted data must be read.
2. The old value must be overwritten with the new value.
3. The old value of the parity data must be read.
4. The old value of the parity data must be overwritten with the new value.

Since these four operations must be performed for every write, the burden is felt most strongly for loads involving many small writes, as the I/Os cannot be amortized over as large an amount of data.

In contrast, systems based on RAID 0+1 simply write the user's data on two separate disks and are only penalized by a factor of two. This disparity, four accesses for small writes instead of two, is termed the "small write problem."

Unfortunately, the performance of online transaction processing (OLTP) systems, a substantial segment of the database storage market, is largely dominated by the performance of small writes. Because of this limitation, many OLTP systems continue to employ the much more expensive option of RAID 0+1, which requires a disk overhead of 100%, as opposed to 20%, which is the typical level of parity overhead for most RAID 5 systems.

Network Appliance's variant of RAID 4 solves this problem by buffering the small writes into memory prior to writing them to disk. Effectively, many small writes are combined into a smaller number of large writes, thus avoiding the small write problem. An excellent discussion of Network Appliance's approach to this problem can be found in *A Storage Networking Appliance* by Dave Hitz and Mike Marchi (http://www.netapp.com/tech_library/3001.html). NetApp's solution combines the performance advantages of RAID 0+1 with the cost advantages of RAID 5. (Indeed, the parity overhead of NetApp RAID 4 is even less than the typical RAID 5 solution—on the order of 7% to 14%.) The results presented in this paper are an indication of the effectiveness of this approach.

Veritas file system with Quick I/O is a software RAID product that uses JBOD hardware. Veritas provides a volume manager that makes it possible to combine JBOD disks into RAID 0+1 or RAID 5 arrays. Further, Quick I/O allows these arrays to be addressed as if they were raw partitions, while maintaining many advantages of a file system. For these reasons, this solution has become extremely popular for storing database files. This paper assumes that a Veritas RAID system represents a typical production configuration and compares the performance of this baseline configuration to a configuration built using Network Appliance filers.

2.3. Storage Networking

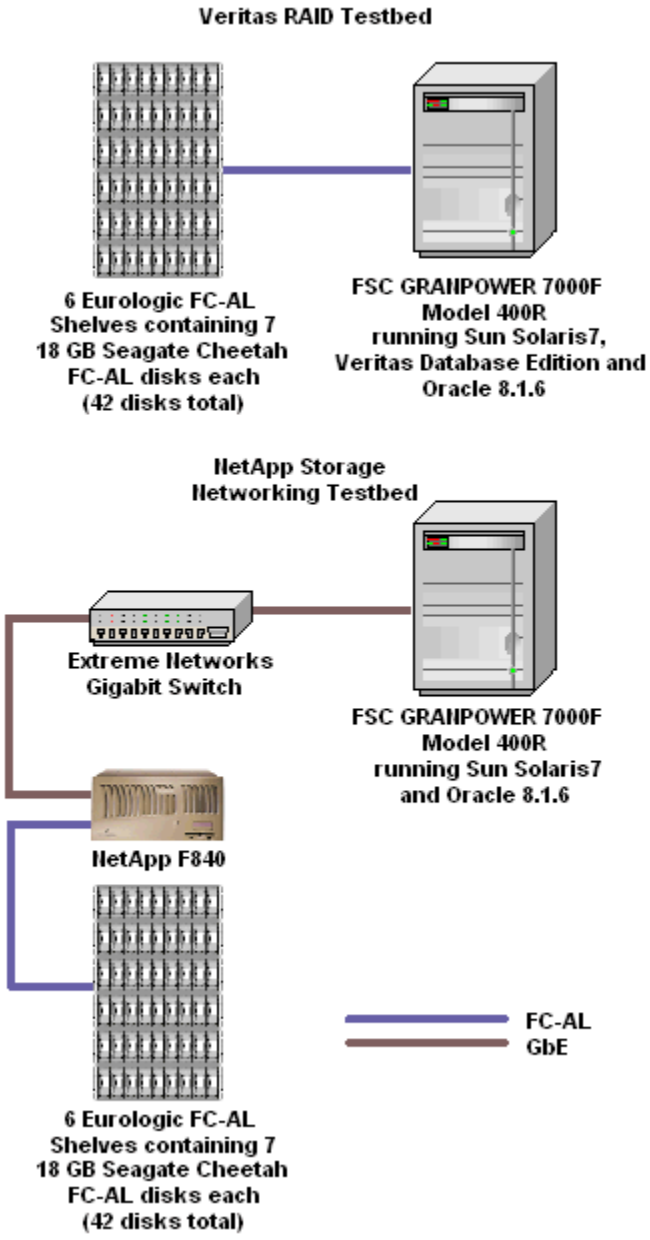
A further evolution in the area of database storage is storage networking. Historically, this has consisted of two areas: NAS (network-attached storage) and SAN (storage area networks). However, many consider these areas to be converging, and the distinction between these two markets is beginning to blur. For example, see "How Convergence Will End the SAN/NAS Debate" by Michael Alvarado and Puneet Pandit, *DM Review*, February 2001 (http://www.dmreview.com/master_sponsor.cfm?NavID=193&EdID=3016). For the purpose of this paper, both of these areas are included in the term "storage networking".

Storage networking is the combination of RAID and some type of networking, either TCP/IP over Ethernet, Fibre Channel, or some other proprietary network technology. The use of networking allows the storage device to be shared by multiple database servers. The functionality advantages of storage networking are clear and have been documented in many places.

Network Appliance storage networking uses TCP/IP over Ethernet. For the purpose of the testing documented in this paper, the physical layer consisted of gigabit Ethernet.

3. Test Description

The testing documented in this paper demonstrates that the performance advantages of NetApp storage networking vs. Veritas RAID are compelling. In order to prove this, we created two realistic configurations, one involving Veritas with Quick I/O, the other involving storage networking using a NetApp filer. Great pains were taken to make these two configurations comparable. In both cases, the disks, controllers, and shelves were identical. Further, these were all currently available, state-of-the-art components. Also, the same database server, with nearly identical settings, was used in both cases. In areas where the settings differed, these were changed in order to achieve the best results on both testbeds. The technical details contained later in this paper provide these settings. The following network diagram contains an overview of the two configurations used.



For the testing documented in this paper, we used an order-entry benchmark to compare the performance of NetApp filers to the Veritas Quick I/O solution. The benchmark's simulated users entered a random mix of five different transactions. These transactions simulated a complete, albeit simple, order-entry and order-fulfillment process. The transactions were processed by an Oracle8i relational database containing records for about 100,000 customer accounts, 100,000 distinct part numbers, and 1,000 warehouses. Although the benchmark was simple, it generated a load on the system that was typical of most order-entry systems. Key features were the use of substantial amounts of CPU time for each transaction and an intense I/O load that mixed random-access reads and writes with sequential I/O and performance-critical recovery logging. A more complete set of benchmark specifications can be obtained through your Network Appliance sales representative.

The server configuration was chosen to be representative of a typical OLTP database server. For example, the most common TCP-C nonclustered result is on a four-way, 4GB system. See http://www.tpc.org/tpcc/default.asp.

4. Test Results

4.1. Statement of Metrics

The following table restates the results of these experiments:

Metric	NetApp F840	Veritas (RAID 0+1)	Veritas (RAID 5)
Average Response Time	.21 sec	.25 sec	.28 sec
Disk Capacity Utilized	24%	43%	21%
Disk Throughput Utilized	50%	68%	65%
Disk Access Times	13 ms	28 ms	21 ms
Disk Volume Creation	< 2 min	3.2 hr	6.5 hr
Database Creation/Load	26.95 hr	50.97 hr	61.30 hr
Transactions per Interval	28,360	24,695	24,085

The F840 filer provided uniformly better response times and better throughput. The throughput differences are particularly notable in the reduced time to load the database.

4.2. Explanation of Metrics

4.2.1. Average Response Times

This metric is the time in seconds that it takes to complete a transaction. The lower the number the better. Response times of 2–3 seconds are usually considered reasonable. A response time below one second is exceptional.

4.2.2. Disk Capacity Utilized

This is the amount of disk space currently in use by the database across all three volumes used in the configuration. A lower number indicates that you have more usable space for future growth.

4.2.3. Disk Throughput Utilized

This metric is the amount of disk utilization measured on the database server. If a disk (or group of disks) measures 100% busy, then that disk has no spare cycles to handle additional I/O requests. A number less than 100% indicates that the disk (or group of disks) could handle more I/O requests, giving you more capacity if future transactions were to increase.

4.2.4. Disk Access Times

This is the amount of time measured in milliseconds that the disks take to respond to an I/O request. A higher number means that the disk group is responding slowly and will increase the average response times the end user will experience. A lower number represents just the opposite: faster disk access with lower transaction response times, which are good for the end user.

4.2.5. Disk Volume Creation

The disk volume creation metric is the amount of time needed to create and bring online a fully functional disk RAID group. This time includes the amount of time needed to enter and execute the commands necessary to create the disk RAID group volumes in question and have them fully ready to accept data.

4.2.6. Database Creation/Load

This metric measures the amount of time needed to bring a fully functional database online. This includes the creation of an empty database, building data dictionary objects, creating the necessary tables and tablespaces to hold the data, loading the actual data, building appropriate indexes, and analyzing all those database objects.

4.2.7. Transactions per Interval

This is the number of OLTP transactions that were completed during the measurement interval. The benchmark runs span a total of ten minutes each. The results listed here are the average results for a five-minute transaction window.

5. Technical Details

This section provides more detail into the specifics surrounding this benchmark comparison.

5.1. Database Server

The database server was a Fujitsu GP7000F Model 400R. This machine was a four-way 296MHz SPARC system equipped with 4GB of physical memory. This system was running Solaris7 with the following patches:

- 107544-03
- 109104-03
- 106541-12

A Qlogic Fibre Channel controller was used in the database server to direct connect this machine to the disks. Seagate 18GB Cheetahs within Eurologic shelves were the disks and shelves used. These disks and shelves were identical to those used in Network Appliance filers, including the F840 used in the storage networking test. Six shelves of these disks were connected to the Qlogic controller.

Veritas Database Edition 2.1.1 for Oracle for Solaris was used to create the necessary RAID 0+1 or RAID 5 volumes. When the databases were created, the files that make up the database were converted to use the Quick I/O feature of Veritas. The Quick I/O feature supports direct I/O and kernel asynchronous I/O and allows databases to access regular files on a VxFS file system as raw character devices, thereby improving transaction processing throughput for Oracle databases.

Here are the parameters that were used in the database server's /etc/system file:

```
* Begin Oracle specific changes
* Semaphores
*-------------------
set shmsys:shminfo_shmmax=8589934592
set shmsys:shminfo_shmmin=1
set shmsys:shminfo_shmmni=900
set shmsys:shminfo_shmseg=300

set semsys:seminfo_semmap=600
set semsys:seminfo_semmni=1000
set semsys:seminfo_semmns=1400
set semsys:seminfo_semmnu=800
set semsys:seminfo_semume=400
set semsys:seminfo_semmsl=1400
set semsys:seminfo_semopm=400

*--------------------
* Message Queue
*--------------------
set msgsys:msginfo_msgmap=1024
set msgsys:msginfo_msgmax=65535
set msgsys:msginfo_msgmnb=65535
set msgsys:msginfo_msgmni=1024
set msgsys:msginfo_msgssz=2048
set msgsys:msginfo_msgtql=1024
* End Oracle specific changes
*
*Increases the size of STREAMS synchronization
set sq_max_size = 1600
set nstrpush = 90
*
set ncsize 8000
set maxusers = 2048
set nfs:nfs3_max_threads = 48
set nfs:nfs3_nra = 10
set priority_paging=1
* vxvm_START (do not remove)
forceload: drv/atf
forceload: drv/pln
forceload: drv/ses
forceload: drv/vxdmp
forceload: drv/vxio
forceload: drv/vxspec
* vxvm_END (do not remove)

* vxfs_START -- do not remove the following lines:
*
* VxFS requires a stack size greater than the default 8K.
```

```
* The following values allow the kernel stack size
* for all threads to be increased to 16K.
*
set lwp_default_stksize=0x4000
set rpcmod:svc_run_stksize=0x4000
* vxfs_END
```

A script called S99netperf was placed in the /etc/rc2.d directory to configure various networking parameters. The script is executed upon reboot and its contents are listed below:

```
case "$1" in
          'start')

               echo "Setting local kernel parameters...\c"
               ndd -set /dev/udp udp_recv_hiwat 65535
               ndd -set /dev/udp udp_xmit_hiwat 65535
               ndd -set /dev/tcp tcp_recv_hiwat 65535
               ndd -set /dev/tcp tcp_xmit_hiwat 65535
               ndd -set /dev/ge instance 0
               ndd -set /dev/ge adv_pauseTX 1
               ndd -set /dev/ge adv_1000autoneg_cap 1
               ndd -set /dev/ge adv_1000fdx_cap 1
               ndd -set /dev/ge instance 1
               ndd -set /dev/ge adv_pauseTX 1
               ndd -set /dev/ge adv_1000autoneg_cap 1
               ndd -set /dev/ge adv_1000fdx_cap 1
               echo " "
               ;;

          'stop')

               echo "$0: No parameters changed."
;;

          *)

               echo "Usage: $0 (start|stop)"
               ;;

          esac
          exit 0
```

5.2. Oracle Settings

Oracle 8.1.6.1 for Solaris (64-bit) was used in all tests. For the most part, the same initialization parameters were used for all tests as well. Several exceptions are noted below.

5.2.1. Common

```
_db_file_noncontig_mblock_read_count = 1
_db_writer_max_writes = 640
_db_writer_chunk_writes = 100
_spin_count = 3000
compatible = 8.1.5.0.0
control_files = $ctrl/ctrl_1,$ctrl/ctrl_2
```

```
cursor_space_for_time = TRUE
db_block_buffers = 750000
db_block_lru_latches = 8
db_block_max_dirty_target = 0
db_block_size = 4096
db_files = 2000
db_file_multiblock_read_count = 1
db_name = oltp1000
distributed_transactions = 0
dml_locks = 200
enqueue_resources = 2000
fast_start_io_target = 0
hash_area_size = 0
hash_join_enabled = false
java_pool_size = 4k
lock_sga = false
log_buffer = 1048576
log_checkpoint_interval = 0
log_checkpoints_to_alert = true
max_rollback_segments = 400
open_cursors = 80
open_links = 0
optimizer_percent_parallel = 0
parallel_automatic_tuning = false
parallel_execution_message_size = 4096
parallel_max_servers = 40
parallel_min_servers = 0
parallel_threads_per_cpu = 8
pre_page_sga = true
processes = 225
recovery_parallelism = 40
replication_dependency_tracking = false
session_cached_cursors = 40
sessions = 225
shared_pool_size = 42000000
sort_area_size = 8192
timed_statistics = true
transactions = 275
transaction_auditing = false
transactions_per_rollback_segment = 1
```

5.2.2. NetApp Storage Networking

The Sun implementation of ASYNC_IO for file systems uses a number of lightweight processes (LWPs). In our testing environment, we found that the kernel overhead of these LWP's was higher than the Oracle overhead for using multiple DB Writers(DBWR). Therefore, we disabled DISK_ASYNC_IO for NFS – mounted databases on Solaris.

```
disk_async_io = false
db_writers = 4
```

5.2.3. Veritas RAID

Asynchronous I/O allows the Oracle DBWR process to schedule multiple I/Os without waiting for the I/O to complete. When the I/O completes, the kernel notifies the DBWR using an interrupt.

Quick I/O supports kernel asynchronous I/O, which reduces CPU utilization and improves transaction throughput. Enabling the following parameter lets Oracle take advantage of asynchronous I/O and avoids having to configure multiple DBWR processes:

```
disk_async_io = true
db_writers = 1
```

5.3. Network Settings

During the software RAID tests there was no need for network connectivity since all tests were performed on the database server machine. However, during the storage networking tests networking was required between the filer and the database server.

There was one gigabit Ethernet card in the database server that was connected to an Extreme Networks Summit4 switch. The F840 filer was also connected via gigabit Ethernet to this same switch. The database server and filer could have been direct connected using a crossover cable. However, we attempted to use a configuration similar to a real-world environment.

5.4. Disk/Volume Settings

5.4.1. NetApp Storage Networking

The filer used one Qlogic FC-AL controller to connect to the 42 disks that were attached to it. These were then configured into volumes using the following commands:

```
vol create oltpnfs1 -r 14 -d 7.0 7.1 7.2 7.3 7.4 7.5 7.6 7.8 7.9 7.10
7.11 7.12 7.13 7.14

vol create oltpnfs2 -r 14 -d 7.16 7.17 7.18 7.19 7.20 7.21 7.22 7.24
7.25 7.26 7.27 7.28 7.29 7.30

vol create oltpnfs3 -r 14 -d 7.32 7.33 7.34 7.35 7.36 7.37 7.38 7.40
7.41 7.42 7.43 7.44 7.45 7.46
```

The following volume options were set:

```
vol options oltpnfs1 minra on
vol options oltpnfs2 minra on
vol options oltpnfs3 minra on
```

5.4.2. Veritas RAID

For those disks directly attached to the database server, Veritas was used to configure the RAID groups.

For RAID 0+1, a RAID 0 group was created using the first seven disks on the FC-AL loop. This RAID 0 group was then mirrored to the next seven disks on the loop. This process was repeated two more times until all 42 disks were configured. The final result was three RAID 0+1 volumes.

For RAID 5, a RAID 5 group was created using the first 14 disks on the FC-AL loop. Two more RAID 5 groups were created until all 42 disks were configured and in use. The final result was three RAID 5 volumes.

These configurations gave each disk farm three mount points over which the database could be built. All the Oracle tablespace files were spread evenly across all three mount points.

NetApp®

Network Appliance, Inc.
495 East Java Drive
Sunnyvale, CA 94089
www.netapp.com