NETAPP TECHNICAL REPORT

Oracle 11*g* Release 1 Performance: Protocol Comparison on Red Hat Enterprise Linux 5 Update 1

Allan K. Wilson, NetApp
April 2009 | TR-3700

INCLUDING DATA ONTAP® 7.2.4 AND DATA ONTAP 7.3 PERFORMANCE COMPARISON

TABLE OF CONTENTS

1 INTRODUCTION AND EXECUTIVE SUMMARY

1.1 INTRODUCTION

NetApp strives to provide up-to-date collateral on its products in an effort to help its customers select the best solutions for their IT infrastructure. This technical report provides updated performance information for Oracle® 11*g*™ R1 on Red Hat Enterprise Linux® 5 Update 1 with NetApp® storage systems, allowing customers to make informed decisions on the optimal storage protocol for their Oracle Database deployment. In addition to comparing storage protocols, the performance improvements Data ONTAP® 7.3 provides in a database environment are also demonstrated.

The protocols used in this technical report include kernel NFS (kNFS), Oracle's new Direct NFS (DNFS), software iSCSI, and FC.

In this report we make the distinction between two different NFS clients. The NFS client provided by the Linux operating system is referred to as kernel NFS or kNFS. The other NFS client, referred to as Oracle Direct NFS or DNFS, is a new NFS client provided by Oracle in 11*g* R1 and runs as part of the Oracle Database. DNFS can only be used to access the Oracle Database files.

Who should read this report? All people who are in the process of selecting a storage protocol for their Oracle Database running on NetApp storage or who are simply curious about how the different protocols compare in performance. This report can also be used as a performance-tuning reference.

1.2 EXECUTIVE SUMMARY

We tested all supported protocols for the Oracle Database to demonstrate the level of performance that each can offer. These protocols include kernel NFS (kNFS), Oracle's new Direct NFS (DNFS), iSCSI using software initiators, and Fibre Channel (FCP). Data ONTAP 7.3, the latest release in the Data ONTAP family, is also compared with the previous release, Data ONTAP 7.2.4, to demonstrate the performance improvements it offers our customers.

An OLTP workload was used for all performance tests. This workload simulated 120 users interacting with 4,000 product warehouses in an order-processing application. The client processes for the OLTP application were executed on a separate application server (client-server mode). The number of Order Entry Transactions (OETs) completed per minute was the primary metric used to measure application throughput. The I/O mix for the OLTP workload was about 65% reads and 35% writes.

The load was configured to saturate host-side resources for the kNFS protocol. This load was then held constant across the other protocols in order to have an apples-to-apples comparison of application throughput. The bottleneck in these tests never existed at the storage layer.

Figure 1 demonstrates that application throughput for all protocols was excellent. Looking first at Data ONTAP 7.2.4 results, application throughput for kNFS was 48,209.80 OETs per minute. The iSCSI protocol saw a 9.4% increase in application throughput over kNFS, delivering 52,764.81 OETs per minute. Oracle's DNFS had an application throughput increase of approximately 26.4% over kNFS, which gave 60,924.08 OETs per minute, the highest on Data ONTAP 7.2.4. FCP was the second highest Data ONTAP 7.2 4 performer with about 21% greater application throughput over kNFS and a delivery of 58,348.51 OETs per minute.

Results for Data ONTAP 7.3 in Figure 1 show an increase in performance across all protocols, but most notably for Oracle DNFS and FCP. Application throughput increased 17.5% (10,213.94 OETs) for FCP, making it the top-performing protocol on Data ONTAP 7.3. Oracle DNFS had an impressive OET increase of 6,362.74 and trailed FCP by only a small margin of 1,275.63 OETs. OETs increased by 1,352.58 for iSCSI and by a modest 89.67 OETs for kNFS.

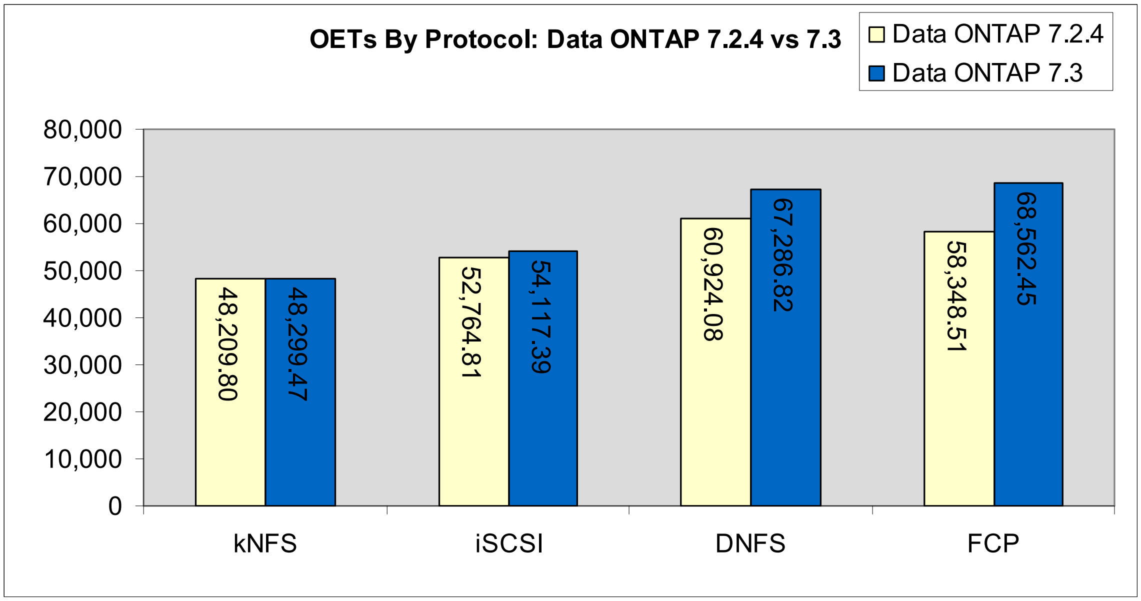


Figure 1) Application throughput comparison: Data ONTAP 7.2.4 vs. 7.3.

The new Oracle DNFS client that runs as part of the Oracle 11*g* R1 Database delivered consistently higher performance than kNFS and is an excellent choice for customers who wish to obtain FCP-like performance at NFS costs for OLTP workloads. No matter which protocol you select for your Oracle Database, Data ONTAP 7.3 provides a high-performance storage platform.

2 CONFIGURATION DETAILS

2.1 GENERIC CONFIGURATION

The test environments used, where possible, the same hardware and similar configurations.

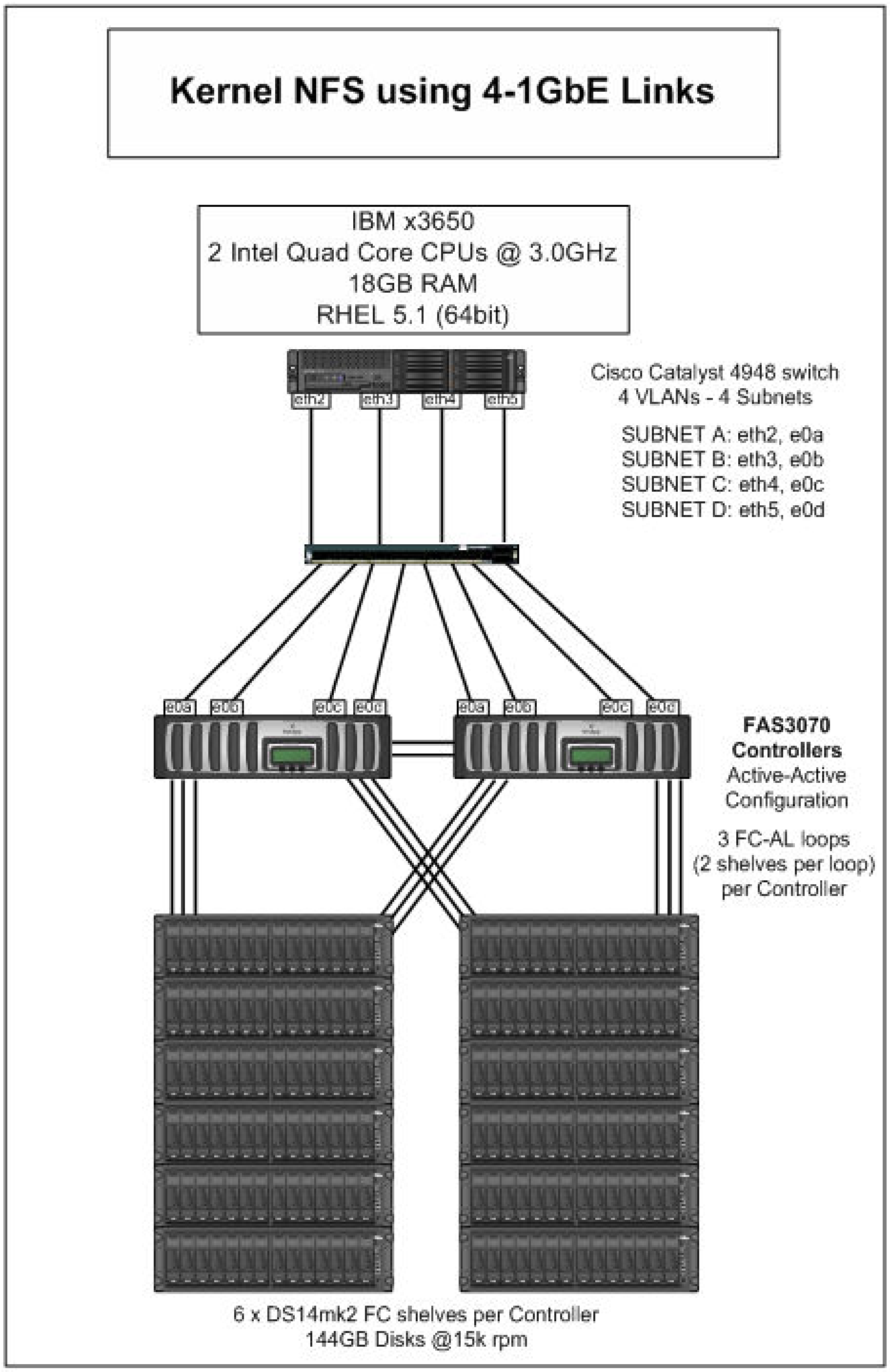
This includes:

- Hardware (APPENDIX A – HARDWARE)
- Storage layouts (APPENDIX B – STORAGE LAYOUTS)
- Oracle initialization parameters (APPENDIX C – ORACLE INITIALIZATION PARAMETERS)
- Linux kernel parameters (APPENDIX D – LINUX KERNEL PARAMETERS)
- Other Linux OS settings (APPENDIX E – OTHER LINUX OS SETTINGS)

When a configuration deviates from the norm or is exclusive to a particular protocol, that configuration setting is explained in detail within the section that describes the test environment for that protocol.

2.2 KERNEL NFS (KNFS)

NETWORK DIAGRAM



STORAGE NETWORK HARDWARE

Database Server	1 Quad Port 1GbE Card (Intel PRO/1000 MT – PCI-E)
NetApp Controllers	4 built-in 1GbE interfaces
Switch	Cisco Catalyst 4948

STORAGE NETWORK CONFIGURATION

Jumbo frames were utilized in this network configuration. An MTU size of 9000 was set for all storage interfaces on the host, for all interfaces on the NetApp controllers, and for the ports involved on the switch.

Four subnets were configured for the storage network. Each storage network interface on the host as well as each interface on the NetApp controller was in its own subnet. It was necessary to use four subnets to distribute the storage network traffic across all four network interfaces. The aggregate interconnect bandwidth of the configuration was effectively 4Gbps.

NFS MOUNTS AND OPTIONS

Figure 2 shows an excerpt from the /etc/fstab file on the Linux host. Here you can see that the nfsdata1 and nfsdata2 volumes were mounted twice on different mount points. This was done to spread out the Oracle data files across all four storage network interfaces on the host and to increase the overall bandwidth. An alternate configuration would be to have two volumes per NetApp controller instead of one for the Oracle data files and then spread the files evenly across all four volumes and mount those four data volumes only once.

```
scotty-1-e0a:/vol/nfsdata1 /oracle/data1  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0
scotty-2-e0b:/vol/nfsdata2 /oracle/data2  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0
scotty-1-e0c:/vol/nfsdata1 /oracle/data3  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0
scotty-2-e0d:/vol/nfsdata2 /oracle/data4  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0
scotty-1-e0a:/vol/nfslog /oracle/log  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0
scotty-2-e0b:/vol/nfsfra /oracle/fra  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0
```

Figure 2) /etc/fstab.

The mount options used come from the NetApp KB article kb7518: Mount options for databases on NetApp NFS, which is available on NetApp's NOW™ (NetApp on the Web) site.

LINUX KERNEL TUNING FOR KNFS

sunrpc.tcp_slot_table_entries = 128

Increasing this parameter from the default of 16 to the maximum of 128 increases the number of in-flight Remote Procedure Calls (I/Os).

Be sure to edit /etc/init.d/netfs to call /sbin/sysctl –p in the first line of the script so that sunrpc.tcp_slot_table_entries is set before NFS mounts any file systems. If NFS mounts the file systems before this parameter is set, the default value of 16 will be in force.

DATA ONTAP TUNING OPTIONS

nfs.tcp.recvwindowsize 262144

Increases the size of the TCP receive window for NFS. The TCP receive window on the client should be adjusted to the same size as the NetApp controller because TCP will reduce the size of the larger TCP receive window to match the size of the smaller TCP window.

We found that increasing this value from the default of 26280 was necessary to reap the performance increase of jumbo frames.

nfs.tcp.xfersize 65536

This is the default setting for this option; make sure that it doesn't get reduced. This option controls the TCP transmit window size for NFS.

STORAGE CONFIGURATION

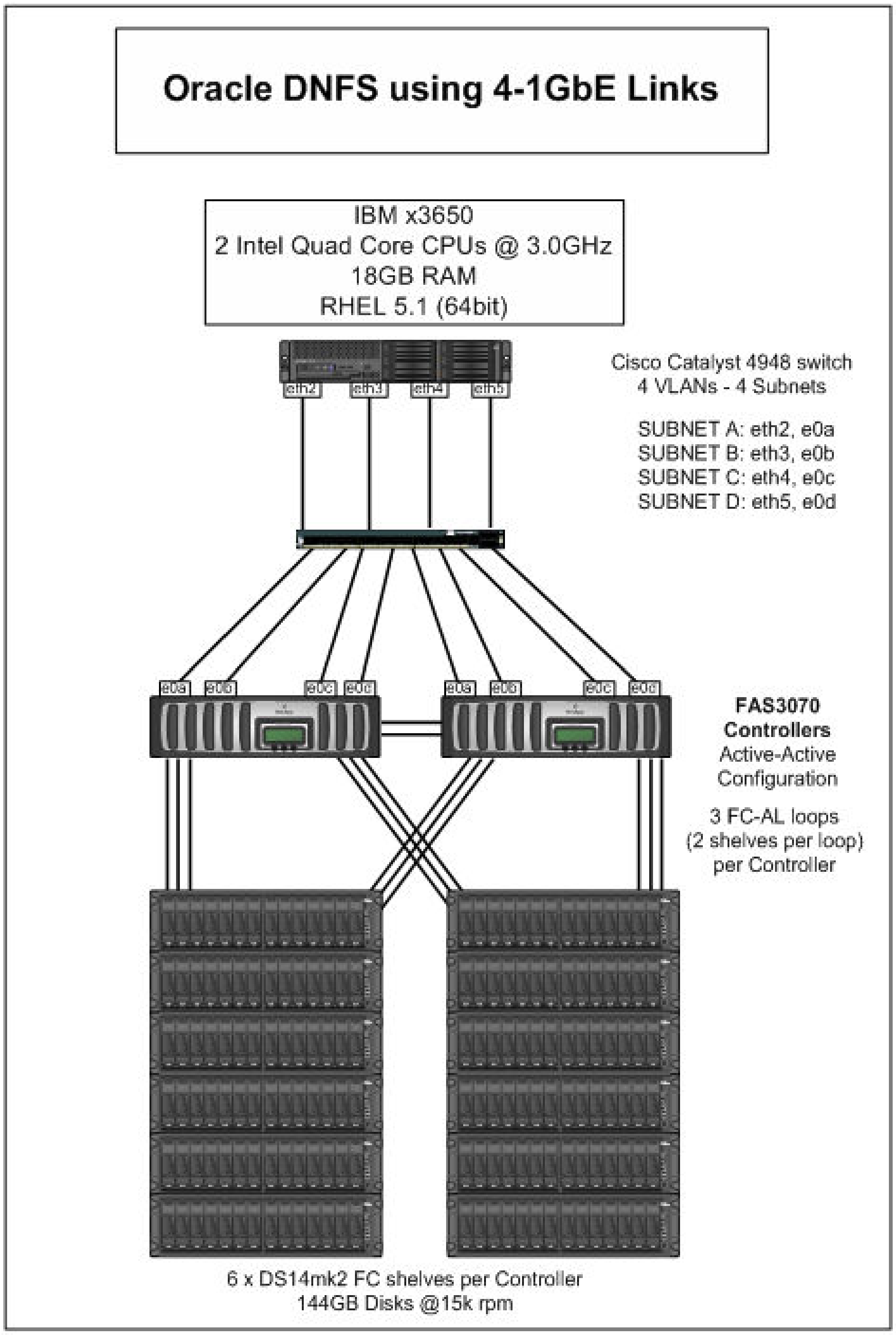
Both Data ONTAP 7.2.4 and Data ONTAP 7.3 were used for these tests.

The storage layout for NFS testing is detailed in the NFS protocol storage layout diagram in APPENDIX B – STORAGE LAYOUTS.

2.3 ORACLE DIRECT NFS (DNFS)

Oracle DNFS is new in Oracle 11*g* R1. DNFS is a NFS client that runs as part of the Oracle Database and is optimized for Oracle Database I/O. DNFS makes it easy to distribute storage network traffic over multiple network interfaces without having to use network bonding on the host or virtual interfaces (VIFS) on the NetApp controllers or without having to spread Oracle data files across numerous mount points, as was done for the kNFS configuration.

NETWORK DIAGRAM



STORAGE NETWORK HARDWARE

Database Server	1 Quad Port 1GbE Card (Intel PRO/1000 MT – PCI-E)
NetApp Controllers	4 built-in 1GbE interfaces
Switch	Cisco Catalyst 4948

STORAGE NETWORK CONFIGURATION

Jumbo frames were utilized in this network configuration. An MTU size of 9000 was set for all storage interfaces on the host, for all interfaces on the NetApp controllers, and for the ports involved on the switch.

Four subnets were configured for the storage network. Each storage network interface on the host as well as each interface on the NetApp controller was in its own subnet. It was necessary to use four subnets to distribute the storage network traffic across all four network interfaces. The aggregate interconnect bandwidth of the configuration was effectively 4Gbps (the same as the kNFS configuration).

NFS MOUNTS AND OPTIONS

Figure 3 shows an excerpt from the /etc/fstab file on the Linux host.

With Oracle DNFS you do not need to spread out the Oracle data files across multiple NFS mount points to spread the storage network traffic over multiple network interfaces. Oracle's DNFS can open multiple network paths for a single NFS mount point. The NFS mount points and network paths are specified in a new configuration file called oranfstab. However, you still must have the NFS mounts specified in the /etc/fstab file, because Oracle cross-checks the entries in this file with the oranfstab file. If any NFS mounts do not match between /etc/fstab and oranfstab, then DNFS will not use those NFS mount points.

For more details, refer to the Oracle Database Installation Guide for Oracle 11*g* R1.

```
scotty-1-e0a:/vol/nfsdata1 /oracle/data1  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0

scotty-2-e0b:/vol/nfsdata2 /oracle/data2  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0

scotty-1-e0a:/vol/nfslog /oracle/log  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0

scotty-2-e0b:/vol/nfsfra /oracle/fra  nfs rw,bg,nointr,rsize=32768,wsize=32768 0 0
```

Figure 3) /etc/fstab.

The mount options used come from the NetApp KB article kb7518: Mount options for databases on NetApp NFS, which is available on the NOW site.

LINUX KERNEL TUNING FOR KNFS

There are no special Linux kernel tuning parameters other than what is shown in APPENDIX D – LINUX KERNEL PARAMETERS.

The parameter **sunrpc.tcp_slot_table_entries** is not required for Oracle DNFS because this parameter only impacts the NFS client provided by Linux. However, NetApp advises you to set this parameter as a precaution, because if Oracle DNFS is unable to mount the Network File System, the Oracle Database will resort to using the Linux NFS client to access the Oracle data files.

ORACLE DNFS CONFIGURATION

Use the oranfstab file to specify the NFS servers, NFS mount points, and the network paths available for each NFS mount. Figure 4 shows the oranfstab file used for this technical report. As you can see, there are four network paths available for each NFS mount.

```
server: scotty-1
path: 10.61.163.101
path: 10.61.164.101
path: 10.61.165.101
path: 10.61.168.101
export: /vol/nfsdata1 mount: /oracle/data1
export: /vol/nfslog mount: /oracle/log

server: scotty-2
path: 10.61.163.102
path: 10.61.164.102
path: 10.61.165.102
path: 10.61.168.102
export: /vol/nfsdata2 mount: /oracle/data2
export: /vol/nfsfra mount: /oracle/fra
```

Figure 4) The oranfstab file located in $ORACLE_HOME/dbs/oranfstab.

DATA ONTAP TUNING OPTIONS

nfs.tcp.recvwindowsize 262144

Increases the size of the TCP receive window for NFS. The TCP receive window on the client should be adjusted to the same size as the NetApp controller, because TCP will reduce the size of the larger TCP receive window to match the size of the smaller TCP window.

We found that it was necessary to increase this value from the default of 26280 to reap the performance increase of jumbo frames.

nfs.tcp.xfersize 65536

This is the default setting for this option; make sure that it doesn't get reduced. This option controls the TCP transmit window size for NFS.

STORAGE CONFIGURATION

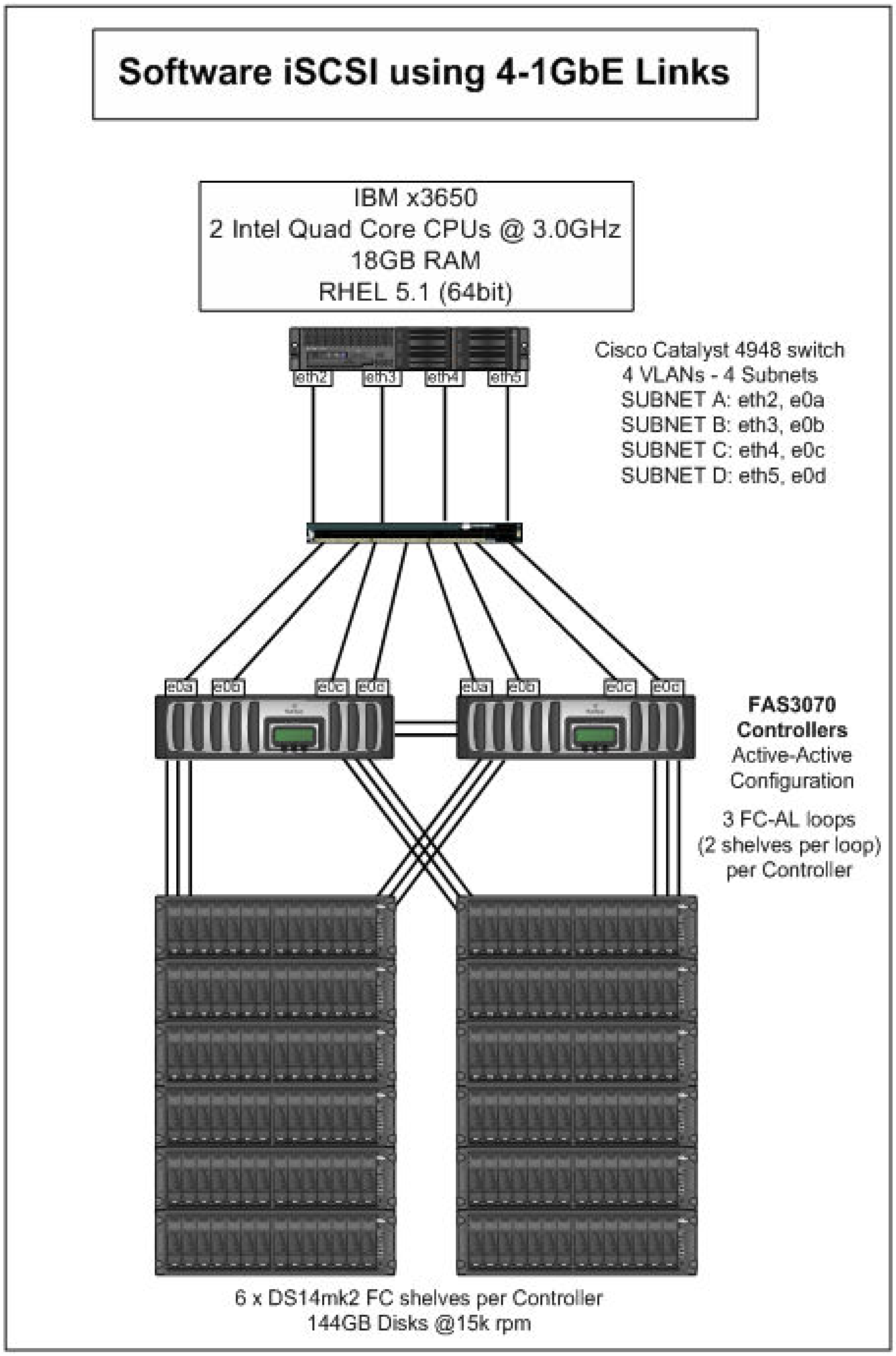
Both Data ONTAP 7.2.4 and Data ONTAP 7.3 were used for these tests.

The storage layout for NFS testing is detailed in the NFS storage layout diagram in APPENDIX B – STORAGE LAYOUTS.

2.4 ISCSI WITH ORACLE ASM AND SOFTWARE INITIATORS

The native Red Hat Linux iSCSI initiator was used in the setup of this environment. There were two iSCSI initiator sessions per NetApp controller. Interfaces e0a and e0b were used on the scotty-1 NetApp controller and interfaces e0c and e0d were used on the scotty-2 NetApp controller. Oracle ASM was used as the file system and volume manager for the LUNs presented from the storage system. A single Oracle ASM DATA disk group was used to evenly spread I/Os across the eight LUNs that were presented from both NetApp controllers.

NETWORK DIAGRAM



STORAGE NETWORK HARDWARE

Database Server	1 Quad Port 1GbE Card (Intel PRO/1000 MT – PCI-E)
NetApp Controllers	4 built-in 1GbE interfaces
Switch	Cisco Catalyst 4948

STORAGE NETWORK CONFIGURATION

Jumbo frames were utilized in this network configuration. An MTU size of 9000 was set for all storage interfaces on the host, for all interfaces on the NetApp controllers, and for the ports involved on the switch.

Four subnets were configured for the storage network. Each storage network interface on the host was in its own subnet as well as each interface on the NetApp controller. Using four subnets was necessary to distribute the storage network traffic across all four network interfaces. The aggregate interconnect bandwidth of the configuration was effectively 4Gbps (the same as kNFS and DNFS).

LINUX KERNEL TUNING FOR ISCSI

There are no special Linux kernel tuning parameters other than what is shown in APPENDIX D – LINUX KERNEL PARAMETERS.

ISCSI CONFIGURATION

NetApp Linux Host Utilities 3.1 was used in conjunction with the native iSCSI initiators. These utilities ease the setup and management of iSCSI on Linux with NetApp and provide the callout program used by DM-Multipath to obtain path priority values.

iSCSI INITIATOR TUNING PARAMETERS:

node.session.queue_depth 128

This setting increased the iSCSI maximum device queue depth.

/sys/lbock/sd<n>/device/queue_depth (SCSI device level setting)

The value in the **queue_depth** file was increased from 32 to 128. This was done for all iSCSI devices.

MULTIPATHING CONFIGURATION

DM-Multipath configuration for iSCSI.

```
devices
{
        device
        {
                vendor "NETAPP"
                product "LUN"
                getuid_callout "/sbin/scsi_id -g -u -s /block/%n"
                prio_callout "/sbin/mpath_prio_ontap /dev/%n"
                features "1 queue_if_no_path"
                hardware_handler "0"
                path_grouping_policy group_by_prio
                failback immediate
                rr_weight uniform
                rr_min_io 128
                path_checker readsector0
        }
```

Figure 5) Multipathing configuration file: /etc/multipath.conf.

ASMLIB CONFIGURATION

Oracle ASMLIB was used to ease the setup and administration of disks for Oracle ASM. The ASMLIB scan order for disks needed to be altered so only multipathed devices would be utilized by ASM.

The changes listed below tell ASMLIB to first search for devices that begin with “dm” (multipathed devices) and exclude devices that begin with “sd” (nonmultipathed devices).

```
# ORACLEASM_SCANORDER: Matching patterns to order disk scanning
ORACLEASM_SCANORDER="dm"
# ORACLEASM_SCANEXCLUDE: Matching patterns to exclude disks from scan
ORACLEASM_SCANEXCLUDE="sd"
```

Figure 6) Oracle ASMLIB configuration file: /etc/sysconfig/oracleasm.

DATA ONTAP TUNING OPTIONS

iscsi.tcp_windowsize 262144

Increased the maximum iSCSI TCP window size from the default of 65536 to match the TCP receive window on the Linux host.

iscsi.max_ios_per_session 256

Increased iSCSI max outstanding I/Os per initiator session from 128 to 256.

STORAGE CONFIGURATION

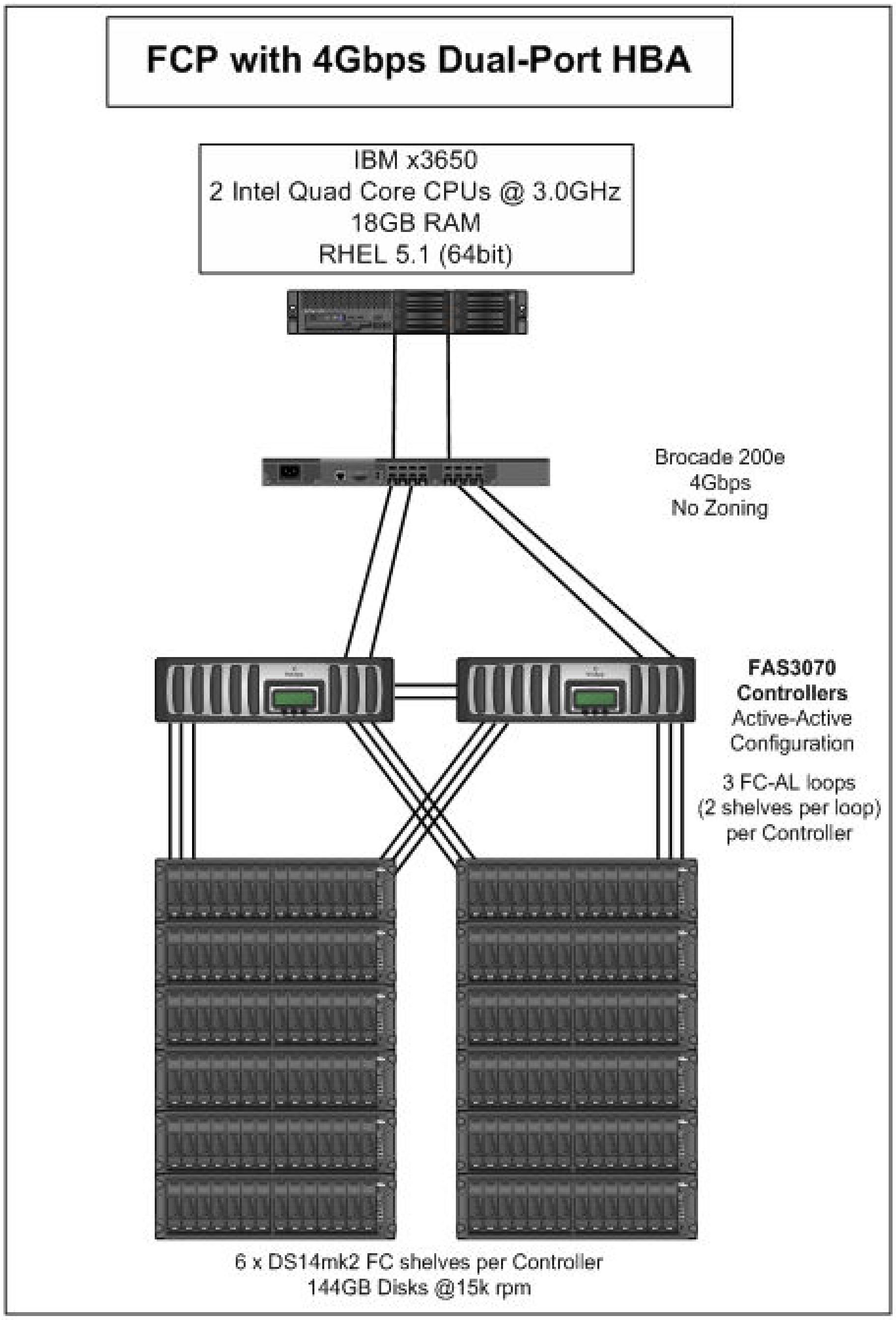
Both Data ONTAP 7.2.4 and Data ONTAP7.3 were used for these tests.

The storage layout is detailed in the block protocol storage layout diagram in APPENDIX B – STORAGE LAYOUTS.

2.5 FIBRE CHANNEL PROTOCOL (FCP) WITH ORACLE ASM

Oracle ASM was used as the file system and volume manager for the LUNs presented from the storage system. A single Oracle ASM DATA disk group was used to evenly spread I/Os across the eight LUNs that were presented from both NetApp controllers.

NETWORK DIAGRAM



STORAGE NETWORK HARDWARE

Database Server	1 Dual Port QLogic 4Gbps HBA (PCI-E)
NetApp Controllers	4 built-in 4Gbps HBAs (2 target, 2 initiator)
Switch	Brocade 200e (4Gbps)

STORAGE NETWORK CONFIGURATION

Both 4Gbps HBA ports from the host were connected to the Brocade 200e switch. The QLogic HBA on the host was set up with the following configuration:

- Frame Size 2048
- Data Rate 4Gbps
- Execution Throttle 256
- Port Down Retry Count 30
- Link Down Timeout 20
- LUNS Per Target 128
- Operation Mode 0 – Interrupt for every I/O completion

To configure the SCSI device queue depth, the ql2xmaxqdepth option was passed to the qla2xxx module.

Below is a sample entry from the /etc/modpobe.conf file.

```
options qla2xxx  ql2xfailover=0  ConfigRequired=0  ql2xmaxqdepth=64
```

The two target HBAs from both NetApp controllers were connected to the Brocade 200e switch. The Data ONTAP CFMODE was set to SINGLE_IMAGE. This is the default mode and is the only mode that supports 4Gbps speeds.

LINUX KERNEL TUNING FOR FCP

There are no special Linux kernel tuning parameters other than what is shown in APPENDIX D – LINUX KERNEL PARAMETERS.

FCP CONFIGURATION

We installed NetApp's FCP Linux Host Utilities 4.1.1. These utilities ease the setup and management of FCP on Linux with NetApp and provide the callout program used by DM-Multipath to obtain path priority values.

MULTIPATHING CONFIGURATION

DM-Multipath configuration for FCP.

```
devices
{
        device
        {
                vendor "NETAPP"
                product "LUN"
                getuid_callout "/sbin/scsi_id -g -u -s /block/%n"
                prio_callout "/sbin/mpath_prio_ontap /dev/%n"
                features "1 queue_if_no_path"
                hardware_handler "0"
                path_grouping_policy group_by_prio
                failback immediate
                rr_weight uniform
                rr_min_io 128
                path_checker readsector0
        }
```

Figure 7) Multipathing configuration file: /etc/multipath.conf.

ASMLIB CONFIGURATION

Oracle ASMLIB was used to ease the setup and administration of disks for Oracle ASM. The ASMLIB scan order for disks needed to be altered so that only multipathed devices are utilized by ASM.

The changes listed below tell ASMLIB to first search for devices that begin with "dm" (multipathed devices) and exclude devices that begin with "sd" (nonmultipathed devices).

```
# ORACLEASM_SCANORDER: Matching patterns to order disk scanning
ORACLEASM_SCANORDER="dm"
# ORACLEASM_SCANEXCLUDE: Matching patterns to exclude disks from scan
ORACLEASM_SCANEXCLUDE="sd"
```

Figure 8) Oracle ASMLIB configuration file: /etc/sysconfig/oracleasm.

STORAGE CONFIGURATION

Both Data ONTAP 7.2.4 and Data ONTAP 7.3 were used for these tests.

The storage layout is detailed in the block protocol storage layout diagram in APPENDIX B – STORAGE LAYOUTS.

3 RESULTS AND ANALYSIS

Before analyzing the test results, it is important to have an understanding of the testing methodology and workload employed.

A consistent testing methodology was utilized across all storage protocols for both Data ONTAP 7.2.4 and Data ONTAP 7.3. This methodology utilized an OLTP workload to demonstrate the capabilities of each protocol. The same database was used for the NFS protocol (kNFS and DNFS) and another duplicate database for all block-based protocols (iSCSI and FC).

3.1 OLTP WORKLOAD DESCRIPTION

The database created for the OLTP workload uses a data model designed for Order Entry transaction processing and is approximately 884GB in size. The OLTP database contains about 6,003 warehouses. For testing across all protocols, a workload that simulates 120 users and 4,000 active warehouses was utilized. The client processes for the OLTP application were executed on a separate application server (client-server mode). The load was configured to saturate host-side resources for the kNFS protocol. For these tests, the host-side resource that became the bottleneck for kNFS was the number of RPC slots and not CPU. The maximum of 128 RPC slots was used for these tests. Please refer to the paragraph titled "LINUX KERNEL TUNING FOR KNFS" in section 2.2 for details about adjusting the number of RPC slots for kNFS. The OLTP load (120 users and 4,000 active warehouses) was held constant across the other protocols in order to have an apples-to-apples comparison for application throughput. There was no bottleneck in these tests on the NetApp storage platform.

A mix of different types of transactions is in use during each OLTP test run. These transaction types include Order Entries, Payments, Order Status, Delivery, and Stock Level.

The number of Order Entry Transactions (OETs) completed per minute is the primary metric used to measure application throughput.

The I/O mix for the OLTP workload is about 65% reads and 35% writes.

3.2 DATA ONTAP 7.2.4 RESULTS

The results in Figure 9 show application throughput by protocol for Data ONTAP 7.2.4. Recall from the OLTP workload description that OETs represent the average number of Order Entry Transactions completed per minute. Oracle's new DNFS protocol had the highest application throughput on Data ONTAP 7.2.4, surpassing even FCP. Oracle DNFS, new in Oracle 11*g*, provides a significant boost in NFS performance for the Oracle Database, offering FCP-like speed for OLTP workloads without the expense of a SAN.

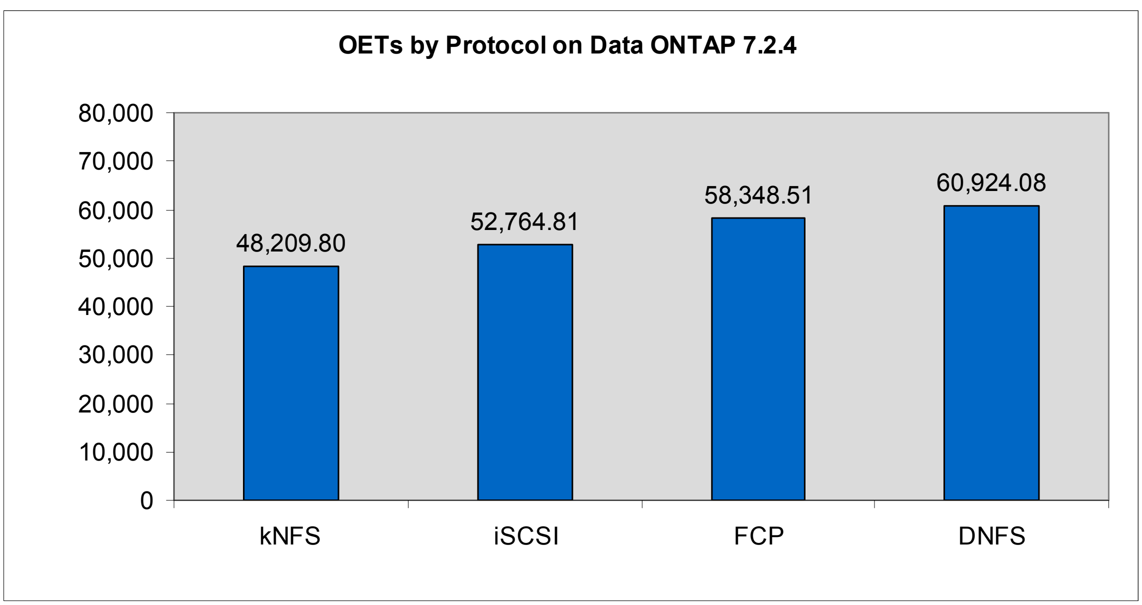


Figure 9) Application throughput by protocol on Data ONTAP 7.2.4.

The FCP protocol with Oracle ASM on Data ONTAP 7.2.4 delivered excellent application throughput and adds the ease of evenly distributing the I/O workload across the LUNs presented from multiple NetApp controllers. iSCSI using software initiators and Oracle ASM delivered the third-highest application throughput and is a cost-effective solution for block-based storage. Kernel NFS still delivers solid and respectable application throughput.

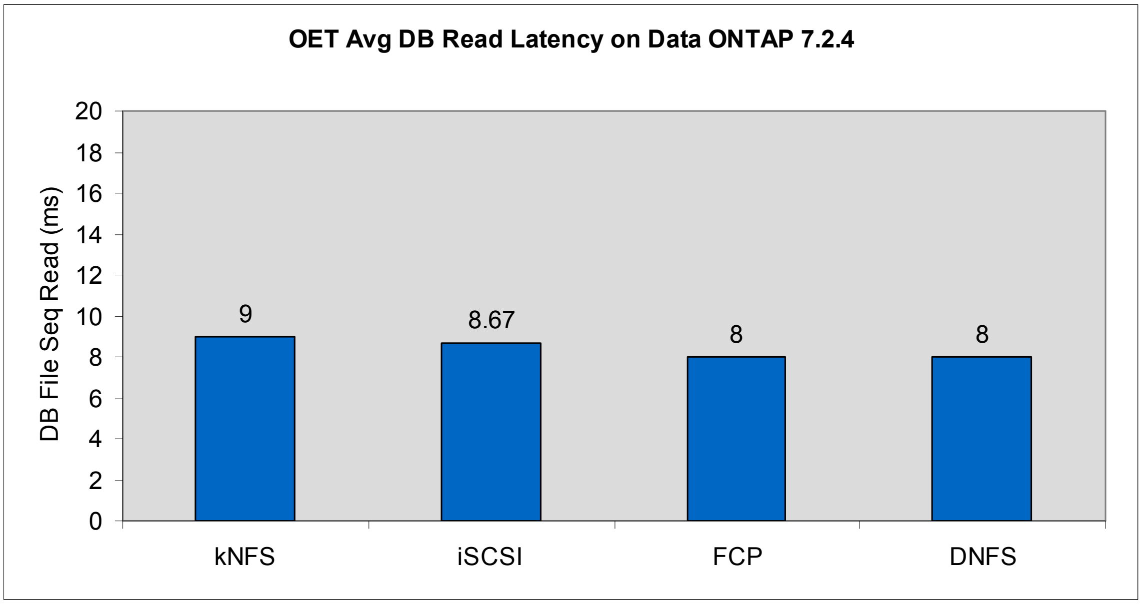


Figure 10) Average Oracle DB single-block read latency by protocol on Data ONTAP 7.2.4 (lower numbers are better).

Figure 10 shows the average DB File Sequential Read wait event reported by the Oracle Database for each protocol. This wait event indicates the amount of time in milliseconds that it took to read a single database block from storage. The top two performers, Oracle DNFS and FCP, were tied at 8ms, followed by respectable iSCSi and kNFS read latencies of 8.76ms and 9ms, respectively (lower results are better).

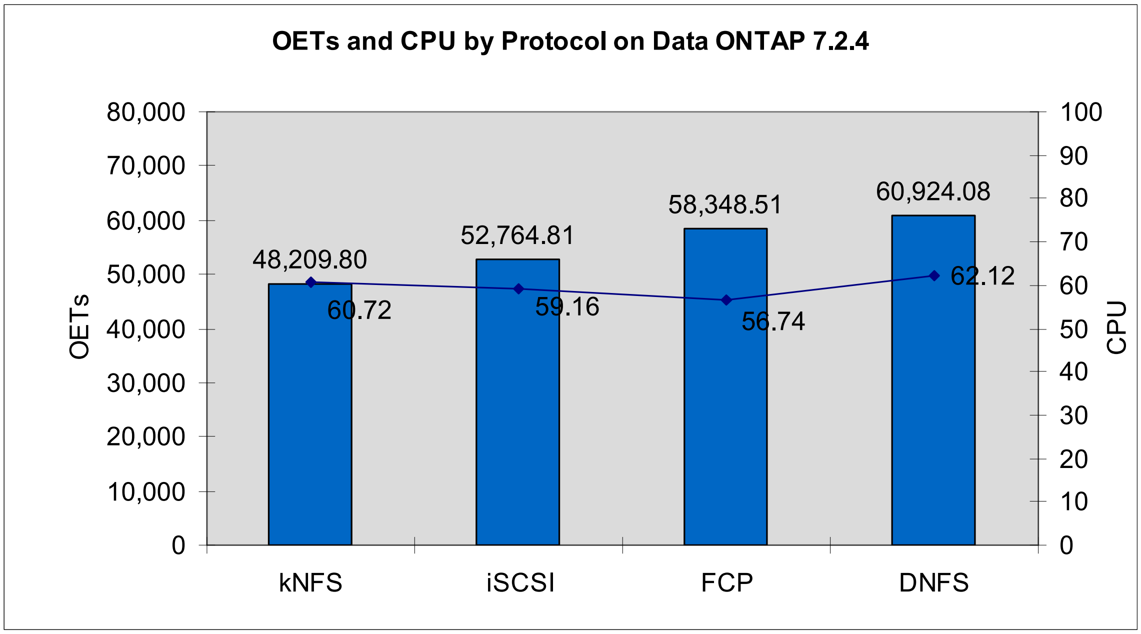


Figure 11) OETs and CPU usage by protocol on Data ONTAP 7.2.4.

Figure 11 shows the average amount of host CPU utilized to generate the corresponding OETs per minute for each protocol. The lowest host CPU consumer is FCP. This is because all storage communications and protocol processing are offloaded onto the FC HBA, freeing up the host CPU. The remaining protocols are all software based and place a little more burden on the host CPU. The Oracle DNFS protocol consumed only 1.4% more host CPU than kNFS despite the fact that average application throughput was more than 26% greater.

The host CPU breakdown by protocol is displayed in Figure 12. kNFS has the largest portion of system CPU as the NFS client runs in the Linux kernel. Notice that Oracle DNFS, although another NFS client, has a much smaller system CPU portion compared to kNFS. This is because the DNFS client runs within the user space of the OS and, as such, the user portion of the CPU usage is larger than kNFS. Another reason the user portion of the CPU is greater for DNFS than kNFS is because DNFS completed more transactions/minute, resulting in a higher number of OETs. As is typical of FCP, it used the smallest amount of system CPU because the storage communication and protocol processing was offloaded onto the HBA.

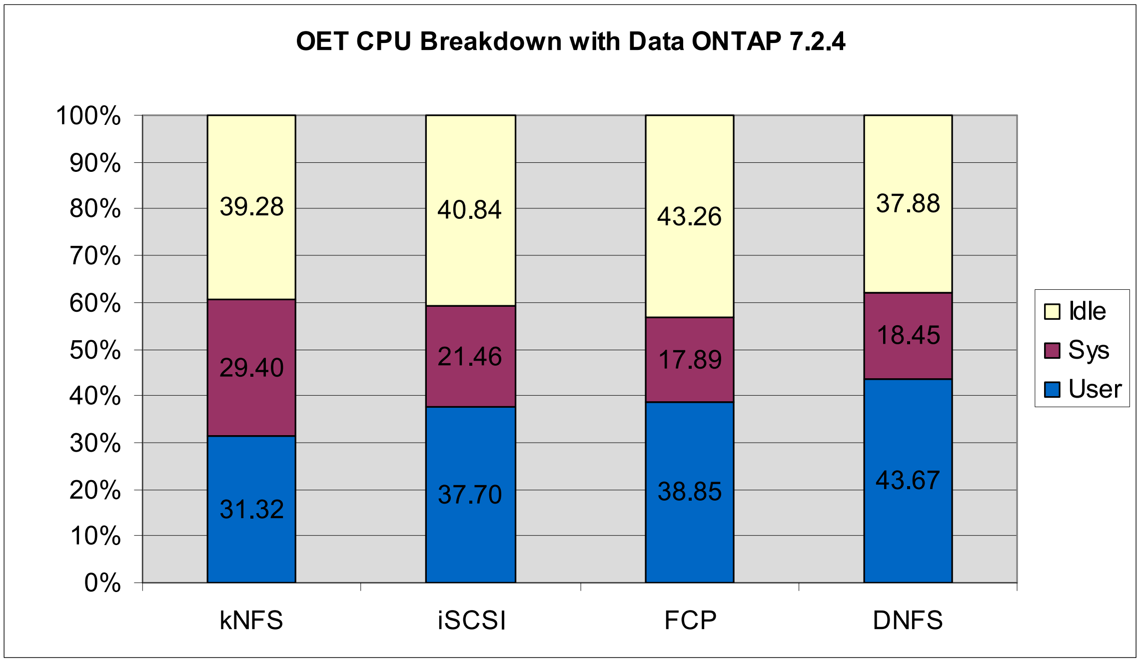


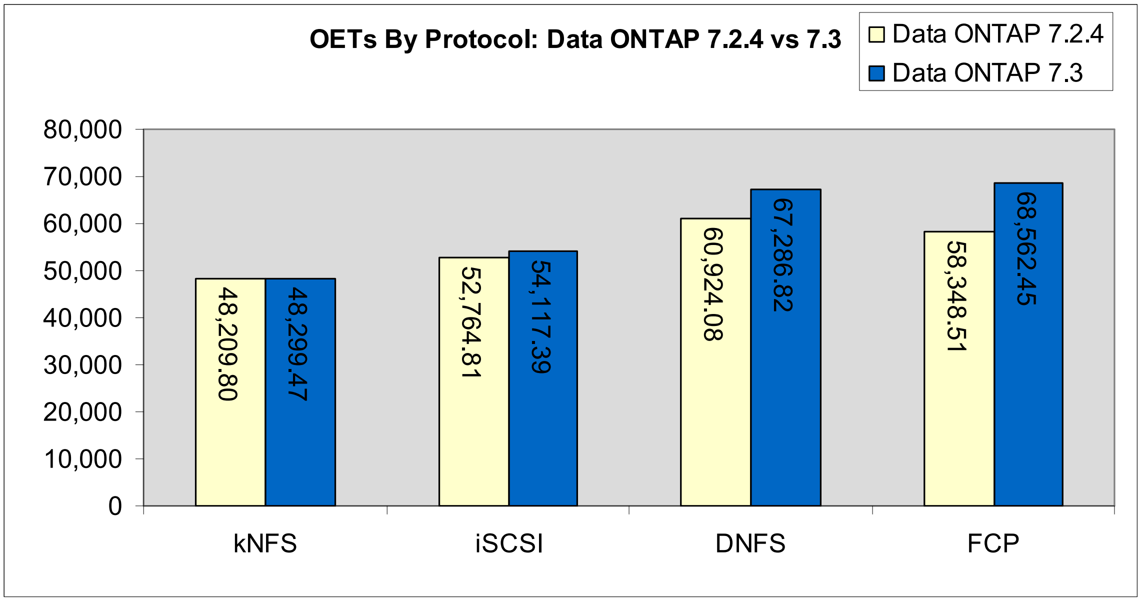
Figure 12) Host CPU usage breakdown by protocol on Data ONTAP 7.2.4.

3.3 DATA ONTAP 7.3 RESULTS

Data ONTAP 7.3, the latest release in the Data ONTAP 7G family, offers many new features and enhancements. These include:

- Slightly larger aggregates (15% or more)
- Faster takeover/giveback (60 seconds or less)
- Nondisruptive upgrade enhancements
- New V-Series features
- Networking/Performance enhancements
- Performance Acceleration Module (PAM)
- FlexCache™ enhancements
- CIFS enhancements
- NFSv4 production ready
- MultiStore® enhancements
- Data protection and deduplication

In addition to these enhancements, our test results showed that Data ONTAP 7.3 provides a performance boost for OLTP workloads on Oracle. Figure 13 displays the increased application throughput achieved in our testing. The workload level (120 users and 4,000 warehouses) and the Oracle Database used for Data ONTAP 7.3 are exactly the same as those used for Data ONTAP 7.2.4.



`

Figure 13) Application throughput comparison: Data ONTAP 7.2.4 vs. Data ONTAP 7.3.

All protocols tested enjoyed some measure of performance improvement. The FCP and Oracle DNFS protocols saw the biggest gains, respectively. Application throughput increased by 17.5% (10,213.94 OETs) for FCP, making it the top-performing protocol on Data ONTAP 7.3. Oracle DNFS had an impressive OET increase of 6,362.74 and trailed FCP by only a small margin of 1,275.63 OETs. OETs increased by 1,352.58 for iSCSI and by a modest 89.67 OETs for kNFS. The reason for the modest OET increase for the kNFS protocol was the RPC slot bottleneck on the Linux host.

Data ONTAP 7.3 reduced the average Oracle DB File Sequential Read event (single-block read time) for all protocols.

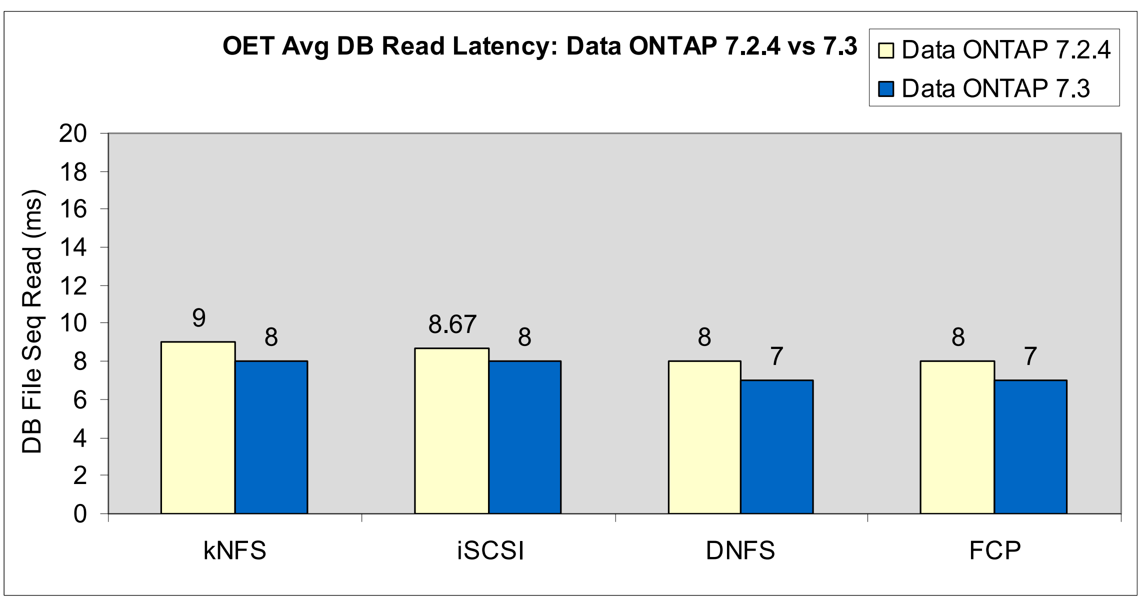


Figure 14) Average Oracle DB single-block read latency by protocol: Data ONTAP 7.2.4 vs. Data ONTAP 7.3 (lower numbers are better).

Figure 14 shows the average DB File Sequential Read wait event reported by the Oracle Database for each protocol. All protocols saw an average reduction of 1ms with Data ONTAP 7.3 for the DB Single Block reads, except for iSCSI, which had an average .67ms reduction.

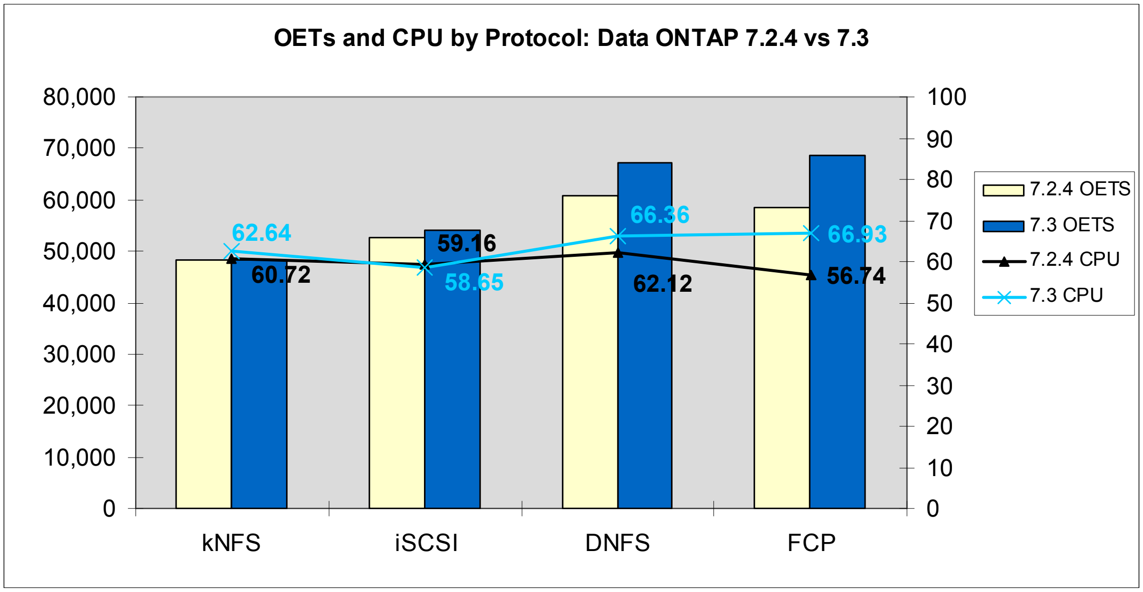


Figure 15) OETs and CPU by protocol: Data ONTAP 7.2.4 vs. Data ONTAP 7.3.

Figure 15 shows the OET and CPU by-protocol comparison between Data ONTAP 7.2.4 and Data ONTAP 7.3. The amount of host CPU for kNFS and iSCSI on Data ONTAP 7.3 is roughly the same as it was on Data ONTAP 7.2.4. This is mainly because of the modest increases in application throughput for these two protocols. Oracle DNFS and FCP saw the biggest increase in CPU usage, 4.24% and 10.19%, respectively, because their application throughput also increased the most. The increase in CPU utilization for DNFS and FC is consistent with their corresponding increase in OET throughput.

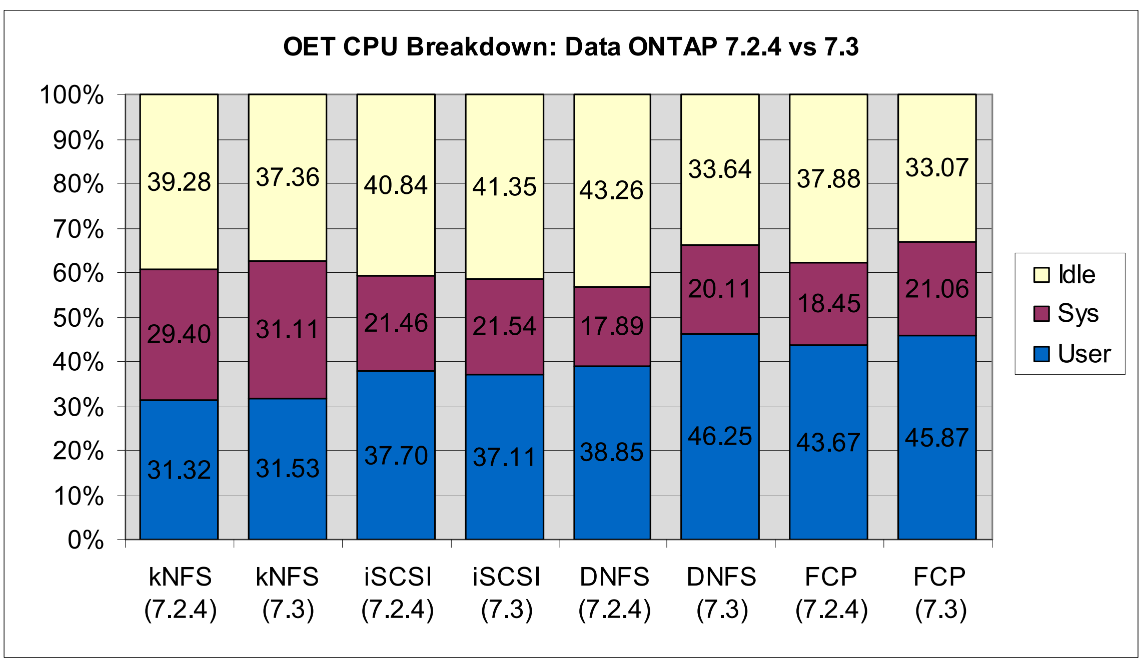


Figure 16) Host CPU usage breakdown by protocol: Data ONTAP 7.2.4 vs. Data ONTAP 7.3.

Figure 16 shows the host CPU usage breakdown. The most notable change in this figure between Data ONTAP 7.2.4 and Data ONTAP 7.3 is that the user and sys CPU components for the DNFS and FC protocols increased for Data ONTAP 7.3. This is because additional application work was taking place because of the better I/O performance delivered by Data ONTAP 7.3.

4 CONCLUSION

NetApp continues to provide leading-edge storage systems that support the NFS (kNFS and Oracle DNFS), iSCSI, and FCP protocols for Oracle deployments. This range of protocol support gives our customers the flexibility to pick which protocol is right for their IT infrastructure and budget. All of the protocols tested performed excellently. Oracle DNFS and FCP are the top two performing protocols in this technical report, which focuses only on OLTP workloads. Oracle DNFS should prove to be an excellent low-cost alternative to FCP for customers deploying OLTP applications.

The release of Data ONTAP 7.3 demonstrates that NetApp continually strives to improve its products by offering new innovative features, increased capacity, and higher performance. The increased application throughput (OETs per minute) and the reduction of Oracle I/O latency demonstrates the performance improvement with Data ONTAP 7.3 across all protocols. No matter which protocol you choose to deploy with your Oracle Database, Data ONTAP 7.3 will provide a high-performance platform that will help maximize your storage investment.

APPENDICES

APPENDIX A – HARDWARE

DATABASE SERVER

IBM 3650
2 Quad Core CPUs @ 3GHz
18GB RAM

NETAPP STORAGE

NetApp FAS3070
2 active/active storage controllers
Each storage controller contained:
 2 Dual Core AMD CPUs @ 1.8GHz
 8GB cache
 512MB NVRAM
 6 Shelves of 144GB 15K RPM FC drives

APPENDIX B – STORAGE LAYOUTS

AGGREGATE CONFIGURATION

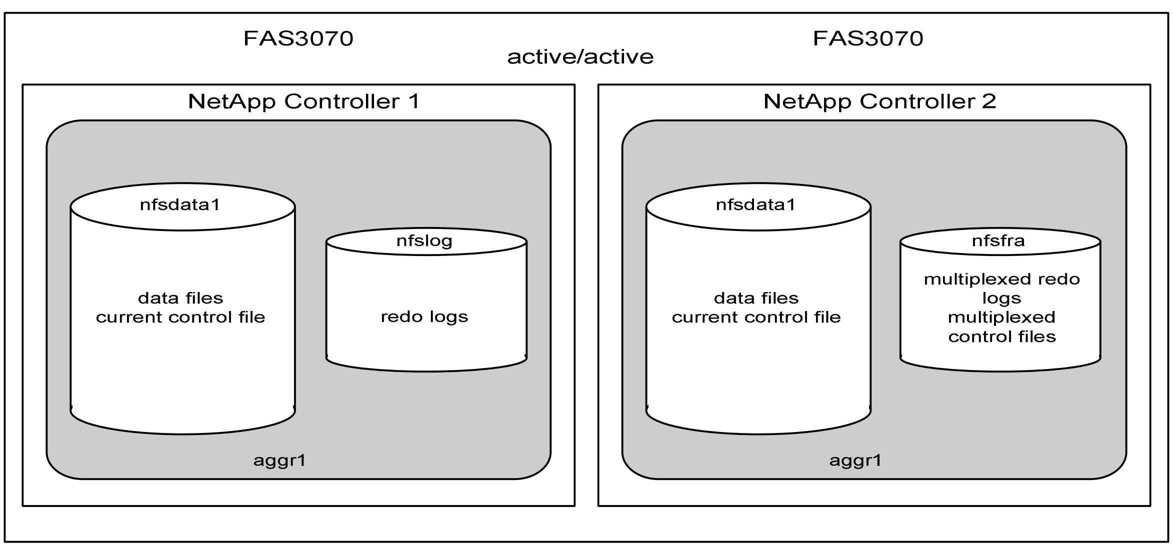
On each of the NetApp storage controllers we created a single aggregate named aggr1 that was dedicated to Oracle Database storage.

Aggregate aggr1 details:

- 80–144GB 15K RPM FC disks
- RAID-DP®
- RAIDSIZE=20

NFS PROTOCOL

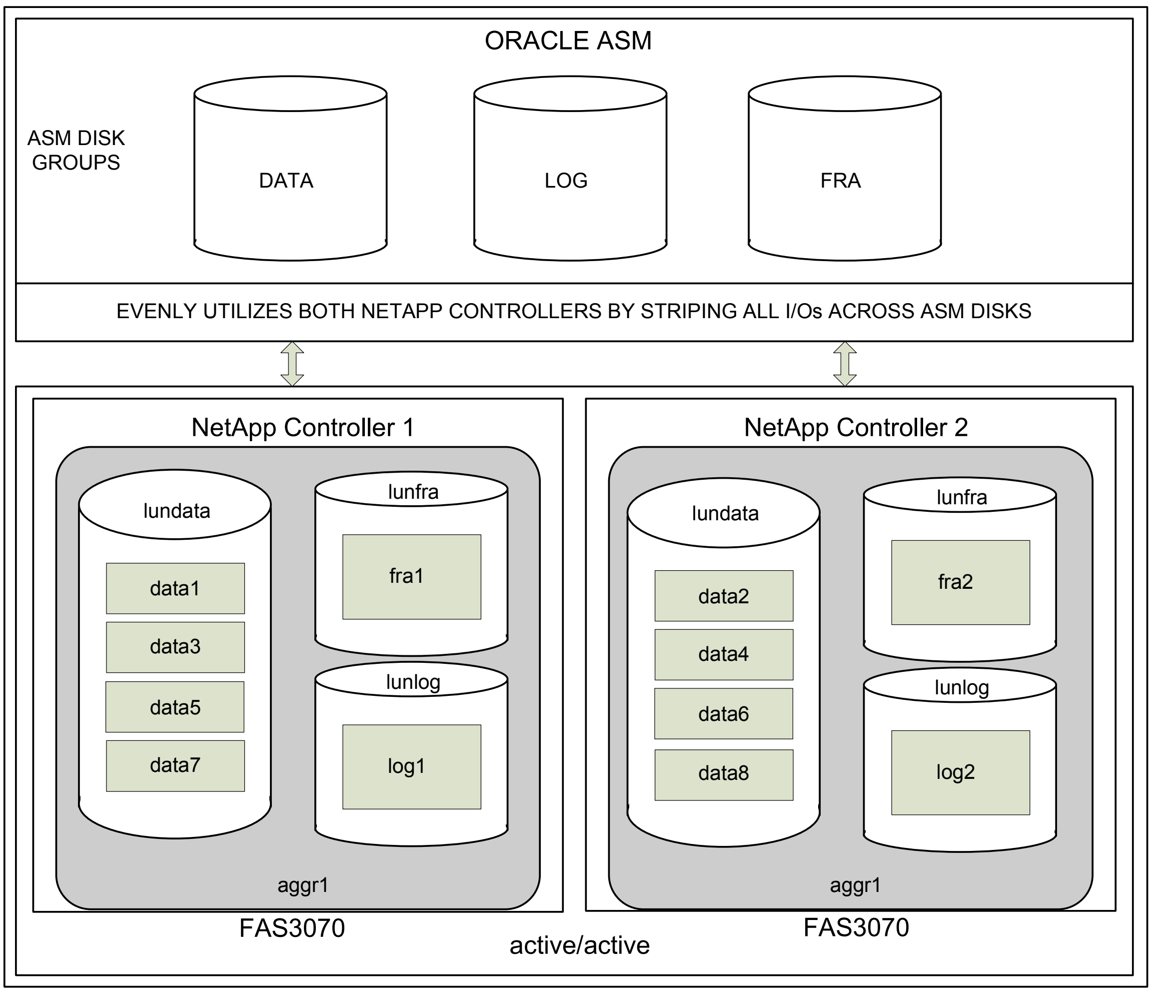
The Oracle data files were evenly distributed across both NetApp storage controllers in the nfsdata<n> volumes. The Oracle online redo logs were also balanced across the two controllers. The primary redo log members were stored on NetApp controller 1 and all multiplexed redo log members were stored on NetApp controller 2.



BLOCK PROTOCOLS

At the database level, the Oracle data files are stored in the ASM DATA disk group. The DATA disk group is comprised of the eight data<n> LUNs located in the lundata volumes. The Oracle online redo logs are stored in the ASM LOG disk group, which is comprised of the two log<n> LUNs located in the lunlog volumes. Multiplexed online redo logs are stored in the FRA ASM disk group, which is comprised of the two fra<n> LUNs located in the lunfra volumes.

All LUNs were presented to Oracle ASM as block devices. That is, a single partition was created on each LUN. Each LUN was partitioned so that the starting sector of the partition began on a 4k boundary to provide proper LUN alignment. Please refer to knowledge base article kb8190: Using partitions on Linux with NetApp LUNs may require alignment for best performance, available on NetApp's NOW Web site.



APPENDIX C – ORACLE INITIALIZATION PARAMETERS

This is a list of the non-default Oracle Initialization Parameters that we set for our performance testing. These values were constant across all the protocols tested.

Parameter Name	Value	Description
_in_memory_undo	FALSE	Make in memory undo for top level transactions
_undo_autotune	FALSE	Enable autotuning of undo_retention
compatible	11.1.0.0.0	Database will be completely compatible with this software version
control_files	/oracle/fra/tpccnfs/control_003	Control file names list
control_files	/oracle/fra/tpccnfs/control_002	Control file names list
control_files	/oracle/data1/tpccnfs/control_001	Control file names list
cursor_space_for_time	TRUE	Use more memory in order to get faster execution
db_16k_cache_size	2147483648	Size of cache for 16K buffers
db_2k_cache_size	0	Size of cache for 2K buffers
db_4k_cache_size	0	Size of cache for 4K buffers
db_block_size	8192	Size of database block in bytes
db_cache_size	10737418240	Size of DEFAULT buffer pool for standard block size buffers
db_file_multiblock_read_count	0	db block to be read each IO
db_files	134	Max allowable # db files
db_name	tpccnfs	Database name specified in CREATE DATABASE
db_recycle_cache_size	0	Size of RECYCLE buffer pool for standard block size buffers
db_writer_processes	4	Number of background database writer processes to start
disk_asynch_io	TRUE	Use asynch I/O for random access devices
dml_locks	500	dml locks - one for each table modified in a transaction
filesystemio_options	SETALL	I/O operations on file system files
log_buffer	33554432	Redo circular buffer size
open_cursors	100	Max # cursors per session
parallel_max_servers	100	Maximum parallel query servers per instance
pga_aggregate_target	524288000	Target size for the aggregate PGA memory consumed by the instance
plsql_optimize_level	2	PL/SQL optimize level
processes	1200	User processes
recovery_parallelism	40	Number of server processes to use for parallel recovery

resource_manager_plan	INTERNAL_PLAN	Resource mgr top plan
sessions	1325	User and system sessions
shared_pool_size	872415232	Size in bytes of shared pool
statistics_level	typical	Statistics level
transactions	1200	Max. number of concurrent active transactions
undo_management	AUTO	Instance runs in SMU mode if TRUE, otherwise in RBU mode
undo_retention	180	Undo retention in seconds
undo_tablespace	undo_1	Use/switch undo tablespace
workarea_size_policy	AUTO	Policy used to size SQL working areas (MANUAL/AUTO)
parallel_execution_message_size	16384	Message buffer size for parallel execution

APPENDIX D – LINUX KERNEL PARAMETERS

This table presents all of the non-default Linux kernel parameters that we used to test all protocols.

Parameter	Value	Description	Notes
kernel.sem	250 32000 100 128	semaphores	11g Install Guide
net.ipv4.ip_local_port_range	1024 65000	Local port range used by TCP and UDP	11g Install Guide
net.core.rmem_default	262144	Default TCP receive window size (Default buffer size)	Improve network performance for IP-based protocols
net.core.rmem_max	16777216	Max. TCP receive window size. (Max. buffer size)	Improve network performance for IP-based protocols
net.core.wmem_default	262144	Default TCP send window size (Default buffer size)	Improve network performance for IP-based protocols
net.core.wmem_max	16777216	Max. TCP send window size (Max. buffer size)	Improve network performance for IP-based protocols
net.ipv4.tcp_rmem	4096 262144 16777216	Autotuning for TCP receive window size (Default and Max. values are overridden by rmem_default rmem_max)	Improve network performance for IP-based protocols
net.ipv4.tcp_wmem	4096 262144 16777216	Autotuning for TCP send window size (Default and Max. values are overridden by wmem_default wmem_max)	Improve network performance for IP-based protocols
net.ipv4.tcp_window_scaling	1	TCP scaling, allows a TCP window size greater than 65536 to be used	This is enabled by default (value 1), make sure that it doesn't get disabled (Value 0).
net.ipv4.tcp_syncookies	0	Disables generation SYN (crypto) COOKIES	Helps to reduce CPU overhead
net.ipv4.tcp_timestamps	0	Disables new RTTM feature	Helps to reduce CPU overhead

		introduced in RFC-1323	Prevents adding 10-byte overhead to TCP header
net.ipv4.tcp_sack	0	Disables selective ack	Helps to reduce CPU overhead

APPENDIX E – OTHER LINUX OS SETTINGS

LINUX SHELL LIMITS FOR ORACLE

Configuration File	Settings
/etc/security/limits.conf	▪ oracle soft nproc 2047 ▪ oracle hard nproc 16384 ▪ oracle soft nofile 1024 ▪ oracle hard nofile 65536
/etc/pam.d/login	▪ session required pam_limits.so
~/.bash_profile	▪ ulimit –n 65536 ▪ ulimit –u 16384 ▪ ulimit –s 32768