Technical Report

Demonstrating the Power and Flexibility of Flash Pool with Oracle Real Application Clusters 11*g*

Chad Morgenstern, NetApp
July 2013 | TR-4195

A Compelling Flash Pool Experience

This technical report demonstrates the benefits of using NetApp® Flash Pool™ technology with online transaction processing (OTLP) workloads. Using Oracle® 11*g* R2 Real Application Clusters (RAC) databases running an OLTP workload on top of NetApp clustered Data ONTAP® 8.1.2, this report shows the superior database performance achievable with a reduced spindle count and the added advantage of a consistent storage failover experience.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction and Executive Summary

1.1 Introduction

NetApp Flash Pool configures solid state drives (SSDs) and hard disk drives (HDDs)—either performance disk drives (often referred to as serial-attached SCSI [SAS] or Fibre Channel [FC]) or capacity disk drives (often called serial ATA [SATA])—into a single aggregate. (An aggregate is the NetApp term for a storage pool.) The SSDs are used to cache data for all volumes that are provisioned on the aggregate.

Provisioning a volume in a Flash Pool aggregate can provide one or more of the following benefits:

- **Persistent fast read response time for large active datasets.** NetApp systems configured with Flash Pool can cache up to 100 times more data than configurations that have no supplemental flash-based cache, and the data can be read 2 to 10 times more quickly than from disk drives. In addition, data cached in a Flash Pool aggregate is available through planned and unplanned controller takeovers, enabling consistent read performance through these events.
- **Provide more HDD operations for other workloads.** Repeat random read and random overwrite operations utilize the SSD cache, enabling HDDs to handle more reads and writes for other workloads, such as sequential reads and writes.
- **Increased system throughput (input/output per second [IOPS]).** For a system where throughput is limited due to high disk drive utilization, adding Flash Pool cache can increase total IOPS by serving a portion of random requests through the SSD cache.
- **HDD reduction.** A storage system configured with Flash Pool to support a given set of workloads typically has fewer of the same type of HDD, and often fewer and lower cost per TB HDDs, than a system not configured with Flash Pool.

Flash Pool is specifically targeted at accelerating repeat random read operations and offloading small-block random overwrite operations (which are a specific class of writes) from HDDs, the type of workload typically generated by OLTP database systems. The small operation size random read-write-read characteristic of OLTP makes it an ideal candidate for Flash Pool.

1.2 Configuration

The configuration environments described in this report consist of the following elements:

- Oracle 11*g* R2 RAC on virtualized Red Hat Enterprise Linux® 5 U9
- NetApp storage systems running clustered Data ONTAP 8.1.2
- VMware VSphere® ESX® 5.0

The study described in this report used Oracle Automated Storage Management (ASM) on top of Oracle Direct Network File System (DNFS) running over 10 Gigabit Ethernet (10GbE) connections. ASM is used to take advantage of the load-balancing properties provided by this feature, allowing IO to be evenly distributed across all storage controllers. DNFS runs as part of the Oracle database software itself and is optimized specifically for database workloads. Therefore, DNFS is used only for accessing the Oracle database files.

1.3 Intended Audience

The target audiences for this report are storage decision makers and administrators investigating the potential deployment of Flash Pool for their Oracle databases using NetApp storage running clustered Data ONTAP 8.1.2.

2 Study Configurations

This section summarizes the configuration details of this study. The study focused on measuring the performance of the Oracle 11*g* R2 RAC databases that generate an OLTP workload, connected to a NetApp FAS6240 dual-controller high availability (HA) system running clustered Data ONTAP 8.1.2.

QLogic 8152 10GbE converged network adapters (CNAs) were used in the database servers connected to NetApp 10GbE unified target adapters (UTAs) installed in the FAS6240 storage nodes. The servers and storage were connected through a Cisco Nexus® 5548 switch.

The performance results of workloads tested on HDD-only aggregates were compared with the result from tests on Flash Pool aggregates. Aggregates consisting on performance drives or capacity drives, and Flash Pool aggregates with both types of drives, were tested.

ASM over DNFS was used between the database servers and the NetApp FAS6240 storage system. Oracle DNFS is an optimized NFS client that provides faster and more scalable access to NFS storage located on network-attached storage (NAS) devices than host-based NFS. DNFS is built directly into the database kernel, bypassing the operating system and generating only the requests required to complete the tasks at hand. DNFS is accessible over TCP/IP.

3 Results and Analysis

Before analyzing the study results, it is important to understand the study methodology and the workload employed. A consistent study methodology was employed for all study cases. This methodology used an OLTP workload to demonstrate the capabilities of the configurations using clustered Data ONTAP 8.1.2.

The tests were designed to simulate a customer environment running typical application workloads. Testing was not done to demonstrate the maximum achievable throughput of each configuration.

3.1 OLTP Workload

The database created for the OLTP workload uses a data model designed for order entry transaction (OET) processing. The OLTP database was approximately 11TB in size and contained approximately 48,000 warehouses.

A mix of different types of transactions was used during each OLTP study run. These transaction types included order entries, payments, order status, delivery, and stock level. The number of OETs completed per minute was the primary metric used to measure application throughput.

The I/O mix for the OLTP workload was approximately 65% reads, 35% writes, and 95% random.

3.2 Detailed Test Results

Figure 1 illustrates the total number of OETs per minute and the average db file sequential read wait time, as reported by the Oracle database for each test run using 15,000 RPM HDDs. The db file sequential read wait time reflects the I/O latency seen at the database layer and indicates the amount of time, in milliseconds, that the database took to read a single database block from storage (that is, a physical read). The workload in Figure 1 was spread evenly across two storage controllers in an HA configuration.

In Figure 1, 6ms physical read response time is breached at 10,000 transactions per minute (TPM) on an aggregate of eighty 15,000 RPM drives. The increase in response time occurred before the HDDs encountered high utilization and while the storage nodes were still lightly utilized. The right-most bar in the figure shows the results for a Flash Pool aggregate that consists of the same eighty 15,000 RPM HDDs plus twenty-two SSDs used as cache. A throughput of 79,000 TPM at 5ms response time was achieved, which is more than nine time higher than the baseline result with only the HDDs (see the left-most bar in Figure 1).

Figure 1) OETs and db file sequential read latencies with performance HDD and Flash Pool aggregates.

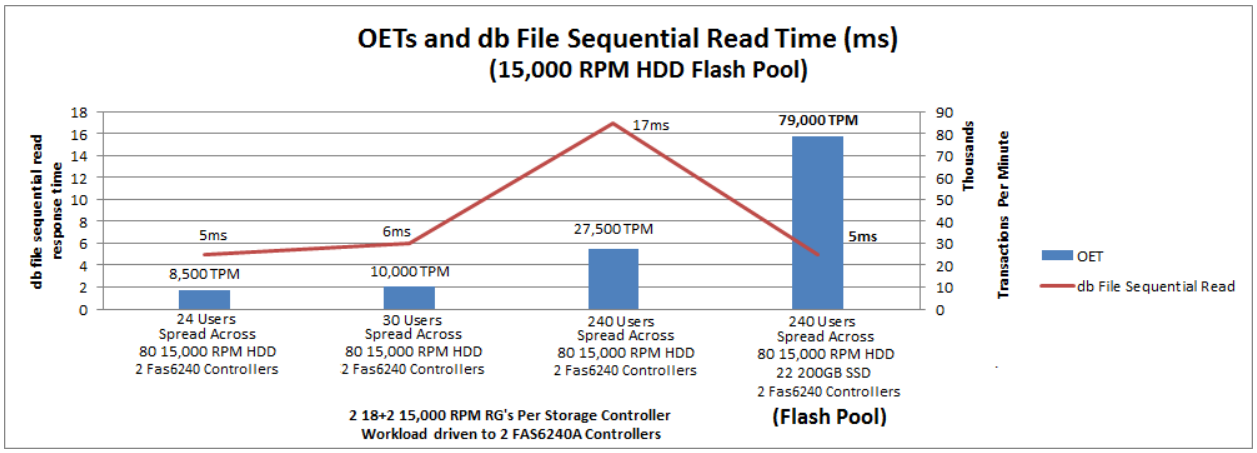
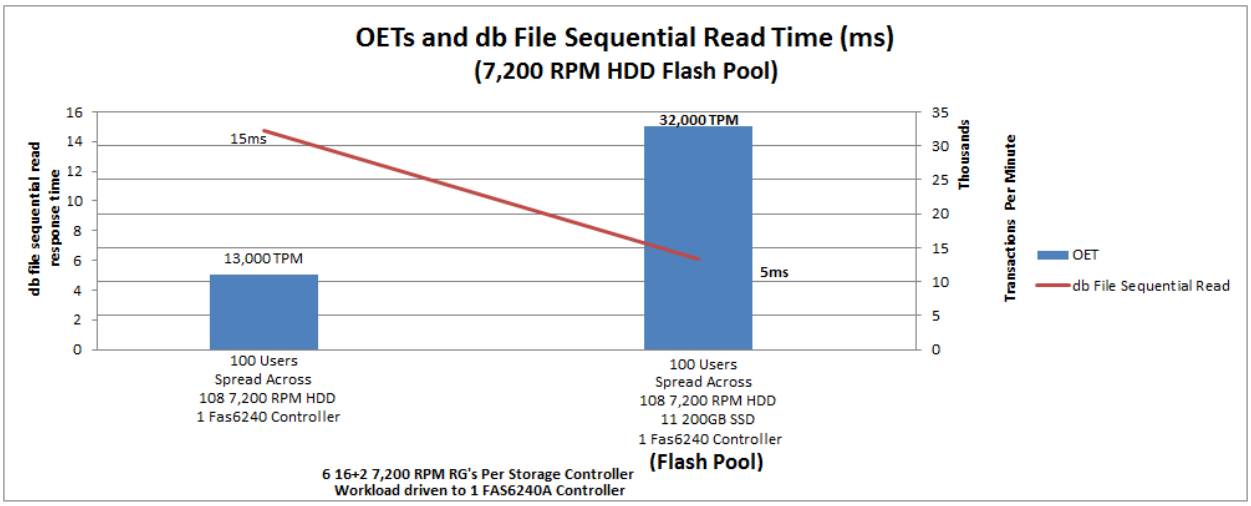


Figure 2 illustrates the total number of OETs per minute and the average db file sequential read wait time as reported by the Oracle database for each test run using 7,200 RPM HDDs. All workloads shown in Figure 2 were served by one node of the HA pair. The Flash Pool aggregate with eleven SSDs and 108 HDDs achieved a throughput of 32,000 TPM at a 5ms response time. Running the same workload on a similarly configured Flash Pool aggregate on the partner node would double the system throughput to 64,000 TPM at the 5ms response time.

Note: The quantity of 7,200 RPM HDDs was greater, and the number of users was lower, than with the 15,000 RPM drive test shown in Figure 1.

Figure 2) OETs and db file sequential read latencies with capacity HDD and Flash Pool aggregates.



3.3 What Makes OLTP an Ideal Candidate for Flash Pool

With the exception of redo and archive logging, OLTP workloads are composed primarily of small block random read and random write workloads. In general, this workload is characterized by a read-write-read pattern with a shifting working set. Consider an automated teller machine (ATM) workload, for example, in which the database must go through a series of Data Manipulation Language (DML) statements consisting of a number of read-write-read operations laid out as follows:

1. Verify account details
2. Accept withdrawal request
3. Check balance
4. Update balance

5. Dispense money
6. Dispense receipt

Figure 3 illustrates that consistently 93% of all reads were served from the SSDs, and of these reads 93% came from the write cache. This exemplifies the read pattern from steady-state OLTP workloads of the read-write-read pattern present in OLTP workloads. Notice that the portion of reads that came from the SSD read cache was roughly equivalent to the total number of reads served from HDD.
As percentage of the total SSD read hits the hits served from read cache is small, however when compared to the total HDD disk operations replaced the read cache actually did a significant amount of work. The testing experimented with all the Flash Pool settings (read cache only, write cache only, and so on) and found that the default settings of read cache and write cache enabled offered the best performance.

Figure 3) Describing the random read working set.

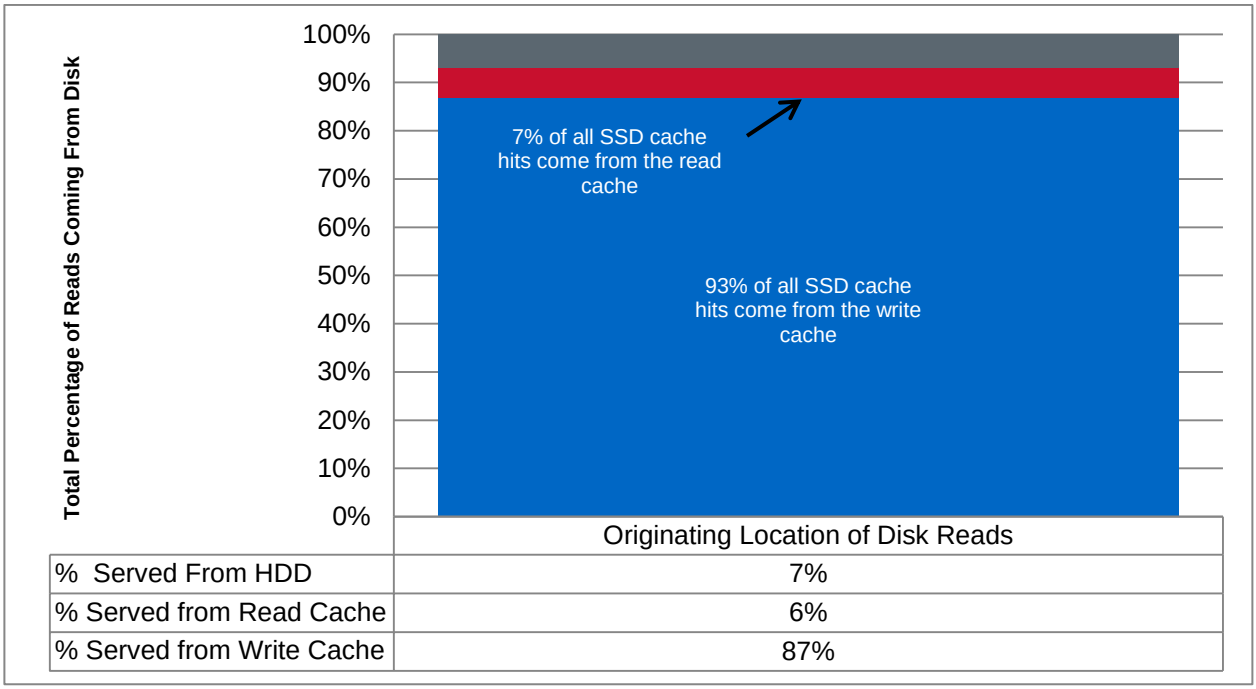
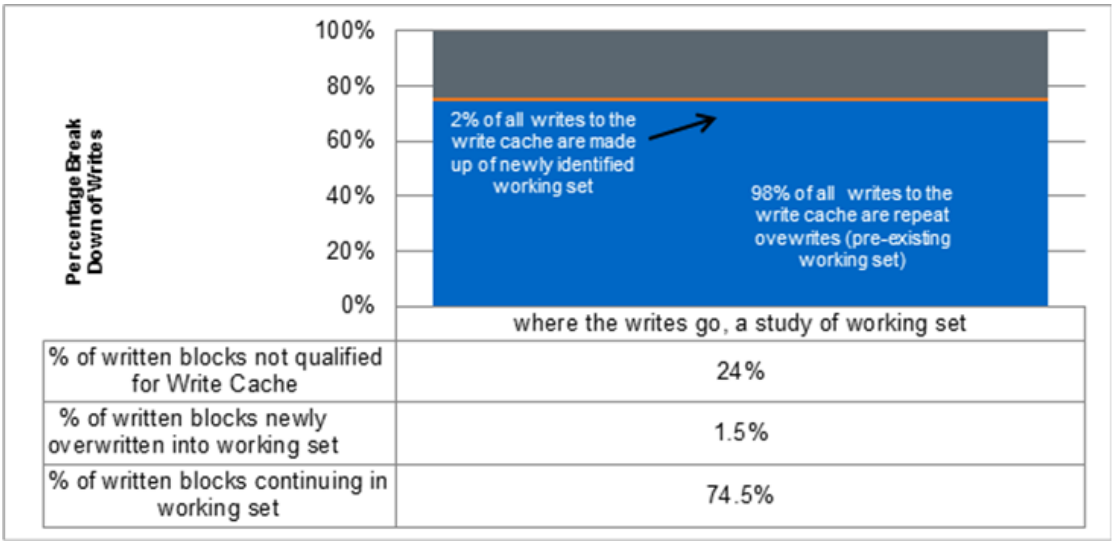


Figure 4 illustrates another notable OLTP and Flash Pool statistic: while three-quarters of all non–redo log writes were written to the write cache instead to the HDDs, 98% of the writes to SSD were written multiple times. This further exemplifies the shifting nature of an OLTP working set. The reads follow the writes until the writes stop occurring and those blocks age out of the cache. The 2% left over represent a new working set or newly hot data.

Figure 4) Describing the random write working set of an 11TB database.

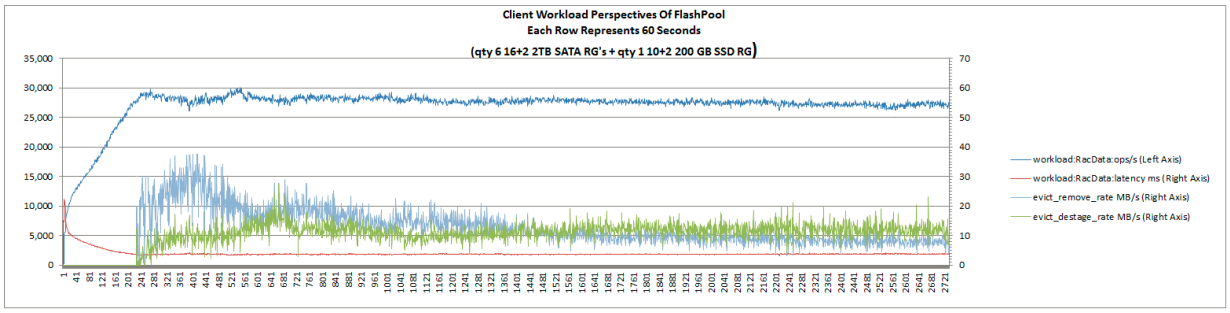


where the writes go, a study of working set	
% of written blocks not qualified for Write Cache	24%
% of written blocks newly overwritten into working set	1.5%
% of written blocks continuing in working set	74.5%

The working set (the portion of the dataset repeatedly accessed) might shift over time, as illustrated in Figure 5. Given a working set that fits within the Flash Pool SSD cache, the less frequently accessed blocks will drop out of the working set as access patterns change. These “cold blocks” will eventually be evicted from cache, or destaged to HDD is they are valid data in write cache, to make room for newer, “hot” data. This was the case with the workload generated in the tests in this report. Eviction and destaging are normal behaviors that do not affect storage latencies, as shown in Figure 5.

An example of this type of workload is a sales inventory system in which a portion of the inventory is composed of hot selling items. In this scenario, the inventory of a portion of the stock is constantly updated. Selling patterns change over time, and items that are no longer hot sellers will fall out of the working set as other more active items replace them.

Figure 5) Consistent throughput and latency with Flash Pool cache evictions and destaging effect of cache destaging on OLTP in a SATA Flash Pool aggregate.



4 Configuration Details

4.1 Oracle RAC Configuration Using Clustered Data ONTAP 8.1.2

The tests for this configuration used standard 10GbE for DNFS over an Intel® dual-port 10GbE network interface card (NIC) in the server connected to NetApp 10GbE unified target adapters (UTAs). The adapters were installed in the FAS6240 controllers and were connected through a Cisco Nexus 5548UP switch.

We used the following configurations:

- DNFS using a 4-node Oracle RAC implementation configured in a vSphere 5.0 environment using four physical servers. In this case, each physical server contained a single VM installed with the Oracle RAC software.

VMware® vCenter™ 5.0 configured on a separate server was used to manage the environment. Red Hat Enterprise Linux 5.9 was installed on each of the VMs created to host the four RAC nodes. Each of the VMs contained a virtual disk for the OS partition that was mapped through the ESX 5.0 NFS stack to the FAS6240 storage system. To access the Oracle database and related files by using DNFS, we configured each of the VMs to directly mount the volumes on the FAS6240 system by using the NFS client in the guest OS to effectively bypass the NFS services of the ESX servers that hosted the RAC nodes.

Network Configurations

Figure 6) Oracle RAC configuration using clustered Data ONTAP 8.1.2 with four physical servers.

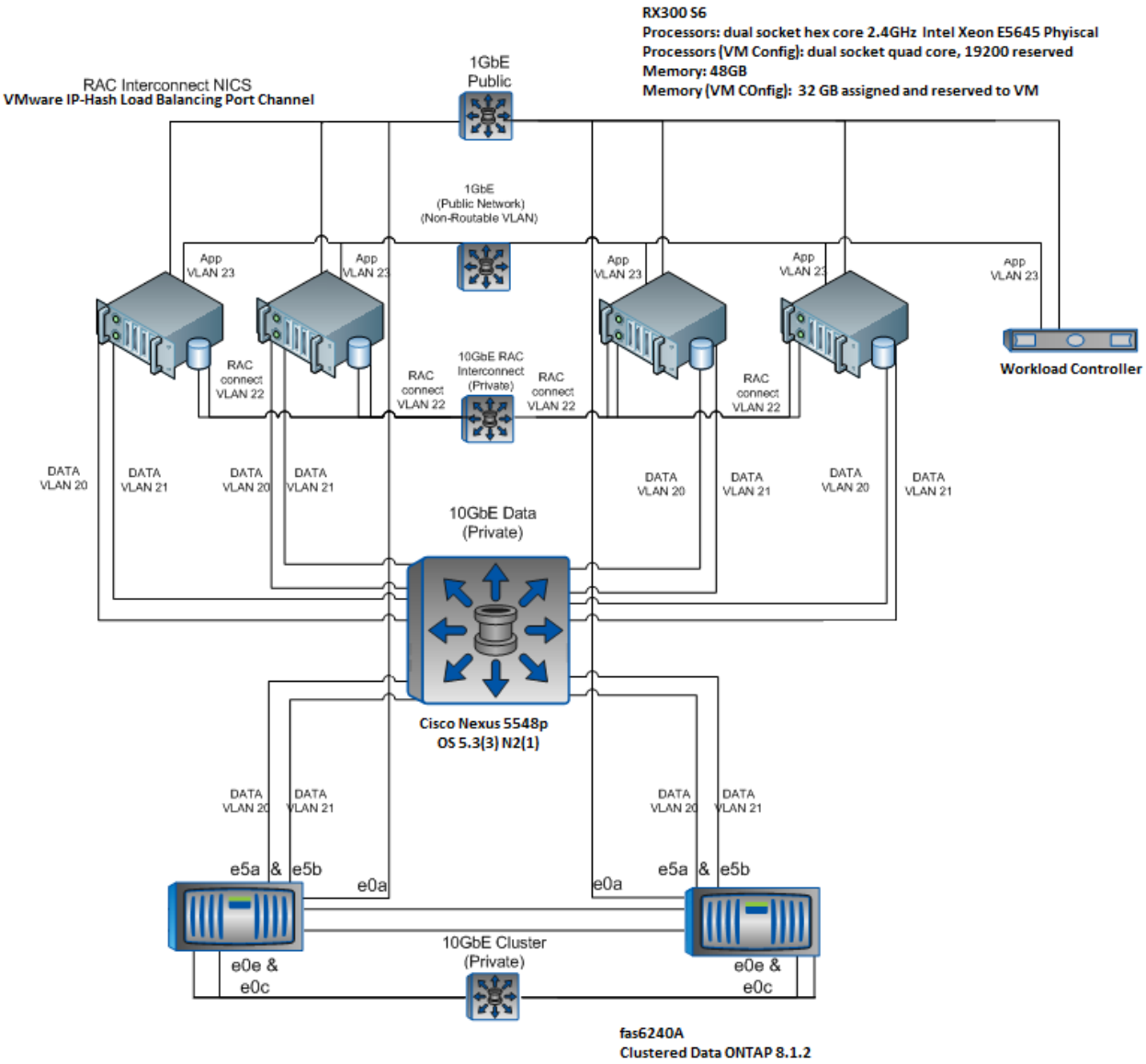


Table 1 lists the hardware for the Oracle RAC configuration using clustered Data ONTAP 8.1.2 with either two or four physical servers using DNFS over 10GbE.

Table 1) Hardware for the Oracle RAC configuration.

Hardware	Description
Database server	4 Fujitsu Primergy RX300 S6 dual hex core CPUs at 2.4GHz, 48GB RAM, RHEL 5 U9, and 2 dual-port 10GbE Intel 82599EB NICs per server
Switch infrastructure for data traffic	1 Cisco Nexus 5548 10GbE switch
Switch infrastructure for Data ONTAP cluster traffic	2 Cisco Nexus 5200 10GbE switches
NetApp controllers	FAS6240A controllers, clustered configuration using multipath high availability (MPHA) and 1 NetApp 10GbE UTA per storage controller
Storage shelves	2 DS4243 SAS shelves per controller, 450GB disks at 15,000 RPM (total disks: 96) 1 DS4246 SAS shelf shared across controllers, 200GB SSD disks (total disks: 24) 5 DS2423 SAS shelves on one controller, 3TB SATA disks at 7,200 RPM (total disks: 120)

Storage Network Configuration

Jumbo frames were used in this network configuration. During these tests, an MTU size of 9,000 was set for all storage interfaces on the host, for all interfaces on the NetApp controllers, and for the ports involved on the switch. Trunking was used to segment Ethernet traffic between the host and the storage nodes.

NFS Mounts and DNFS Configuration

Oracle DNFS path information can be optionally configured in the `orafstab` file. If this file does not exist, DNFS will use the NFS server information from the OS `mounttab` file and use the same IP addresses for filesystem access that are used by the kernel NFS filesystems visable to the use. The tests in this document utilized multipath DNFS which requires the use of an `oranfstab` file to define the different paths to be used.

When conducting tests using Oracle RAC and clustered Data ONTAP 8.1.2, a set of NFS mount points on the RAC database nodes was created that allowed the RAC nodes to access the database files uniformly across the FAS6240 controllers. In clustered Data ONTAP 8.1.2, the NFS mount points on the RAC nodes are specified using a combination of the logical network interface and export junction path that provides access to the different Oracle configuration and database files. The following mount points were defined in the `/etc/fstab` file on the Linux hosts supporting RAC nodes 1 through 4.

RAC Node 1

```
x.x.3.1:/ocrvote1
on/u02/ocr_vote1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.2:/ocrvote2
on/u02/ocr_vote2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.3:/ocrvote3
on/u02/ocr_vote3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.1:/orabin1
on/u01/app(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraData1
on/u03/oradata1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraData2
on/u03/oradata2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraData3
on/u03/oradata3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
```

```
x.x.3.7:/OraData4
on/u03/oradata4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraLog1
on/u04/oralog1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraLog2
on/u04/oralog2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraLog3
on/u04/oralog3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraLog4
on/u04/oralog4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraTemp1
on/u05/oratemp1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraTemp2
on/u05/oratemp2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraTemp3
on/u05/oratemp3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraTemp4
on/u05/oratemp4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
```

RAC Node 2

```
x.x.3.1:/ocrvote1
on/u02/ocr_vote1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.2:/ocrvote2
on/u02/ocr_vote2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.3:/ocrvote3
on/u02/ocr_vote3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.2:/orabin2
on/u01/app(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraData1
on/u03/oradata1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraData2
on/u03/oradata2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraData3
on/u03/oradata3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraData4
on/u03/oradata4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraLog1
on/u04/oralog1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraLog2
on/u04/oralog2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraLog3
on/u04/oralog3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraLog4
on/u04/oralog4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraTemp1
on/u05/oratemp1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraTemp2
on/u05/oratemp2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraTemp3
on/u05/oratemp3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraTemp4
on/u05/oratemp4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
```

RAC Node 3

```
x.x.3.1:/ocrvote1
on/u02/ocr_vote1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.2:/ocrvote2
on/u02/ocr_vote2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.3:/ocrvote3
on/u02/ocr_vote3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.4.3:/orabin3
on/u01/app(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraData1
on/u03/oradata1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraData2
on/u03/oradata2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraData3
on/u03/oradata3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
```

```
x.x.3.7:/OraData4
on/u03/oradata4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraLog1
on/u04/oralog1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraLog2
on/u04/oralog2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraLog3
on/u04/oralog3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraLog4
on/u04/oralog4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraTemp1
on/u05/oratemp1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraTemp2
on/u05/oratemp2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraTemp3
on/u05/oratemp3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraTemp4
on/u05/oratemp4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
```

RAC Node 4

```
x.x.3.1:/ocrvote1
on/u02/ocr_vote1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.2:/ocrvote2
on/u02/ocr_vote2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.3:/ocrvote3
on/u02/ocr_vote3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.4.4:/orabin4
on/u01/app(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraData1
on/u03/oradata1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraData2
on/u03/oradata2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraData3
on/u03/oradata3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraData4
on/u03/oradata4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraLog1
on/u04/oralog1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraLog2
on/u04/oralog2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraLog3
on/u04/oralog3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraLog4
on/u04/oralog4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.4:/OraTemp1
on/u05/oratemp1(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.5:/OraTemp2
on/u05/oratemp2(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.6:/OraTemp3
on/u05/oratemp3(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
x.x.3.7:/OraTemp4
on/u05/oratemp4(rw,bg,hard,rsize=65536,wsize=65536,nfsvers=3,actimeo=0,nointr,timeo=600,tpc)
```

The following are excerpts from the `oranfstab` file that was used during the RAC performance study to define the mount points for use by the Oracle DNFS client. The `oranfstab` file excerpts that follow were used by all four RAC nodes.

```
server: CmodeDBN1
path: x.x.3.4
path: x.x.4.4
export: /OraData1 mount: /u03/oradata1
export: /OraLog1 mount: /u04/oralog1
export: /OraTemp1 mount: /u05/oratemp1
#
server: CmodeDBN2
path: x.x.3.5
path: x.x.4.5
export: /OraData2 mount: /u03/oradata2
```

```
export: /OraLog2 mount: /u04/oralog2
export: /OraTemp2 mount: /u05/oratemp2
#
server: CmodeDBN3
path: x.x.3.6
path: x.x.4.6
export: /OraData3 mount: /u03/oradata3
export: /OraLog3 mount: /u04/oralog3
export: /OraTemp3 mount: /u05/oratemp3
#
server: CmodeDBN4
path: x.x.3.7
path: x.x.4.7
export: /OraData4 mount: /u03/oradata4
export: /OraLog4 mount: /u04/oralog4
export: /OraTemp4 mount: /u05/oratemp4
```

For more information about DNS configuration, refer to the Oracle Database Installation Guide for Oracle 11*g* R2.

Clustered Data ONTAP Tuning Options

Clustered Data ONTAP 8.1.2 was used for all tests. For details on the storage layout for DNFS study, see Figure 7 in “Storage Layouts for All Study Configurations” in the appendix.

5 Conclusion

NetApp is very well known for providing high-performance storage systems for Oracle database environments. With the advent of Flash Pool, NetApp continues to develop leading edge technologies that provide the ability to scale out both capacity and performance in support of NetApp customers’ current and future Oracle database environments.

The study results presented in this report infer that NetApp Flash Pool provides excellent performance and reduced disk drive configuration for OLTP database environments.

Appendix

Hardware

Table 2) Database server hardware specifications for Oracle RAC study.

Component	Details
System type	Fujitsu Primergy RX300 S6
Operating system	RHEL 5 U6
Processor	2 quad core CPUs at 2.4GHz
Memory	48GB
Oracle database version	11.2.0.3.0
Network connectivity	1 dual-port 10GbE Intel 82599EB NIC

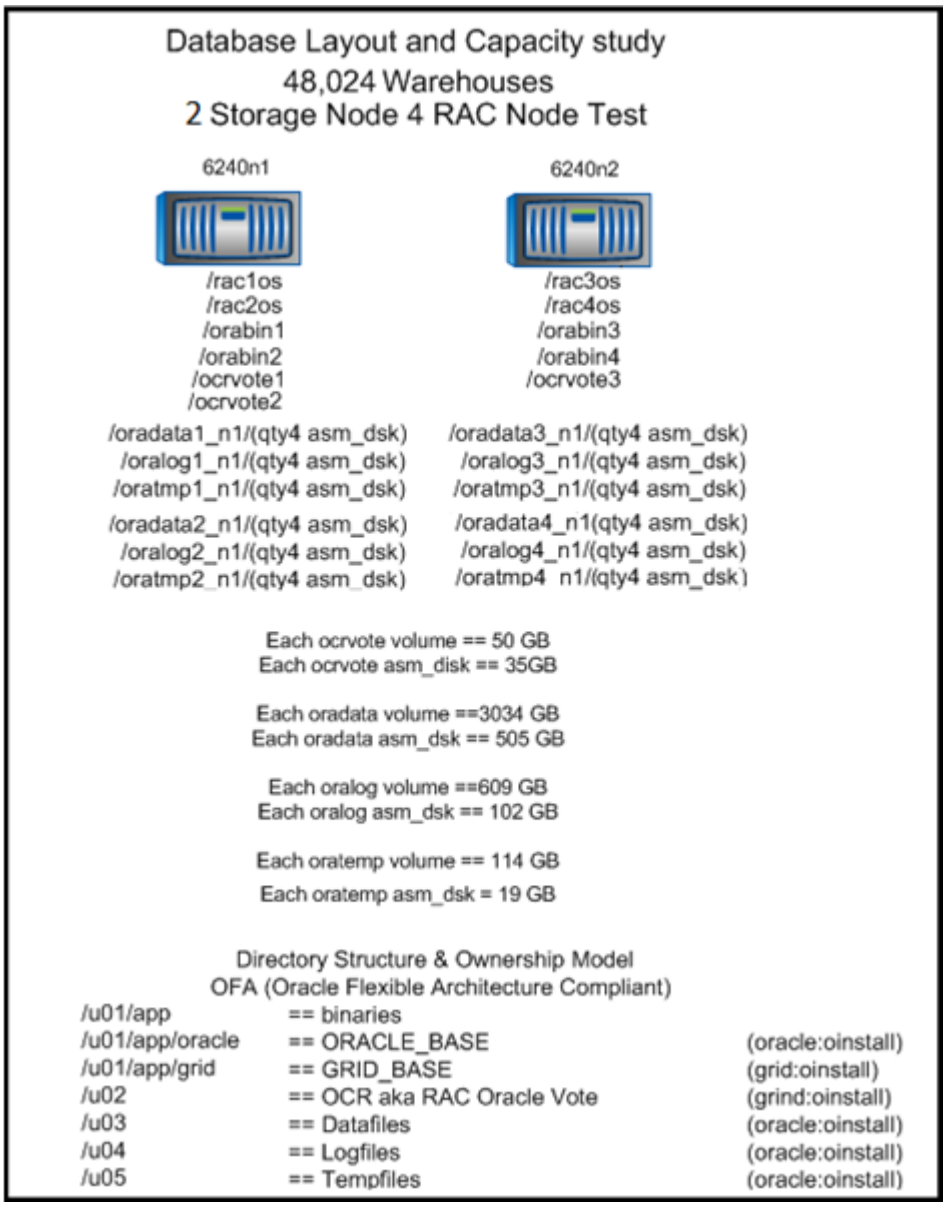
Table 3) NetApp FAS6240A storage system specifications.

Component	Details (in Each Storage Controller)
System type	NetApp FAS6240A HA pair
Operating system	Clustered Data ONTAP 8.1.2
Processor	2 dual-core Intel E5540 CPUs at 2.53GHz
Memory	48GB
NVRAM	4GB
Disks	2 DS4243 SAS shelves per controller, 450GB disks at 15,000 RPM 1 DS4246 SAS shelf shared across controllers, 200GB SSD disks 5 DS2423 SAS shelves on one controller, 3TB SATA disks at 7,200 RPM
Network devices	10GbE UTA

Storage Layouts for All Study Configurations

Figure 7) DNFS volume layout for Oracle RAC study using clustered Data ONTAP 8.1.2.shows the layout used for DNFS. The Oracle data files are distributed evenly across both NetApp storage nodes in the /oradata1_n1 /oradata1_n2 as well as /oradata3_n1 /oradata3_n2 NFS volumes. The Oracle online redo logs are also balanced across the two controllers. The primary redo log members are stored on NetApp controller 1, and all multiplexed redo log members are stored on NetApp controller 2.

Figure 7) DNFS volume layout for Oracle RAC study using clustered Data ONTAP 8.1.2.



Oracle Initialization Parameters for RAC Configuration Study

Table 4) Oracle initialization parameters for RAC configuration study.

Parameter Name	Value	Description
_allocate_creation_order	FALSE	During allocation, files should be examined in the order in which they were created
_in_memory_undo	FALSE	Make in-memory undo for top-level transactions
_undo_autotune	FALSE	Enable autotuning of undo_retention
cluster_database	TRUE	RAC database
compatible	11.2.0.0.0	Database is completely compatible with this software version
control_files	/u08/oradata1/control_001 , /u09/oradata2/control_002	Control file location
db_16k_cache_size	5368709120	Size of cache for 16K buffers
db_2k_cache_size	268435456	Size of cache for 2K buffers
db_4k_cache_size	268435456	Size of cache for 4K buffers
db_block_size	8192	Size of database block, in bytes
db_cache_size	536870920	Size of DEFAULT buffer pool for standard block size buffers
db_files	500	Maximum allowable number of database files
db_name	tpcctest	Database name specified in CREATE DATABASE
dml_locks	500	
filesystemio_options	setall	Use both asynchronous IO and direct IO
instance_number	1	DB instance number
log_buffer	16777216	Redo circular buffer size
open_cursors	1024	Maximum number of open cursors
parallel_max_servers	100	Maximum parallel query servers per instance
plsql_optimize_level	2	Optimization level that will be used to compile PL/SQL library units

Parameter Name	Value	Description
processes	1000	User processes
recovery_parallelism	40	Number of server processes to use for parallel recovery
remote_listener	orarac-scan:1521	Name of remote listener
remote_login_passwordfile	EXCLUSIVE	The password file can be used by only one database
sessions	1536	User and system sessions
shared_pool_size	4294967296	Size, in bytes, of shared pool
spfile	/u08/oradata1/spfile.ora	Location of SPFILE
statistics_level	typical	Statistics level
thread	1	Database threads
undo_management	AUTO	If TRUE, instance runs in SMU mode; otherwise in RBU mode
undo_retention	10800	Undo retention in seconds
undo_tablespace	undo_1	Use or switch undo tablespace

Linux Kernel Parameters for RAC Configuration Study

Table 5) Linux nondefault kernel parameters for RAC configuration study.

Parameter Name	Value	Description
sunrpc.tcp_slot_table_entries	128	Maximum number of outstanding async I/O calls
kernel.sem	240 32000 100 128	Semaphores
net.ipv4.ip_local_port_range	9000 65500	Local port range used by TCP and UDP
net.core.rmem_default	262144	Default TCP receive window size (default buffer size)
net.core.rmem_max	16777216	Maximum TCP receive window size (maximum buffer size)
net.core.wmem_default	262144	Default TCP send window size (default buffer size)
net.core.wmem_max	16777216	Maximum TCP send window size (maximum buffer size)
fs.file-max	6815744	Maximum number of file handles that the Linux kernel allocates
fs.aio-max-nr	1048576	Maximum number of allowable

Parameter Name	Value	Description
		concurrent requests
net.ipv4.tcp_rmem	4096 262144 16777000	Receive buffer size parameters
net.ipv4.tcp_wmem	4096 262144 16777000	Send buffer size parameters
net.ipv4.tcp_window_scaling	1	Allow increase of the TCP receive window size
net.ipv4.tcp_syncookies	0	TCP syncookies disabled
net.ipv4.tcp_no_metrics_save	1	Don't save characteristics of the last connection in the flow cache
net.ipv4.tcp_moderate_rcvbuf	0	Disable TCP received buffer autotuning
fs.file-max	6815744	Maximum number of file handles that the Linux kernel allocates
fs.aio-max-nr	1048576	Maximum number of allowable concurrent requests

Other Linux OS Settings

Table 6) Linux shell limits for Oracle.

Configuration File	Settings
/etc/security/limits.conf	oracle hard nofile 65536

Acknowledgments

Special thanks to the following people for their contributions:

- Keith Griffin, NetApp
- Neto Antonio Jose Rodrigues, NetApp
- Saad Jafri, NetApp
- Skip Shapiro, NetApp

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

NetApp provides no representations or warranties regarding the accuracy, reliability, or serviceability of any information or recommendations provided in this publication, or with respect to any results that may be obtained by the use of the information or observance of any recommendations provided herein. The information in this document is distributed AS IS, and the use of this information or the implementation of any recommendations or techniques herein is a customer's responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. This document and the information contained herein may be used solely in connection with the NetApp products discussed in this document.

Go further, faster®

www.netapp.com