Technical Report

Oracle Databases on ONTAP

Jeffrey Steiner, NetApp
July 2017 | TR-3633

Important

Consult the Interoperability Matrix Tool (IMT) to determine whether the environment, configurations, and versions specified in this report support your environment.

NetApp®

TABLE OF CONTENTS

LIST OF TABLES

1 Introduction

NetApp® ONTAP® is a powerful data-management platform with native capabilities that include inline compression, nondisruptive hardware upgrades, and the ability to import a LUN from a foreign storage array. Up to 24 nodes can be clustered together, simultaneously serving data through Network File System (NFS), Common Internet File System (CIFS), iSCSI, Fibre Channel (FC), and Fibre Channel over Ethernet (FCoE) protocols. In addition, NetApp Snapshot® technology is the basis for creating tens of thousands of online backups and fully operational database clones.

In addition to the rich feature set of ONTAP, there is a wide variety of user requirements, including database size, performance requirements, and data protection needs. Known deployments of NetApp storage include everything from a virtualized environment of approximately 6,000 databases running under VMware ESX to a single-instance data warehouse currently sized at 996TB and growing. As a result, there are few clear best practices for configuring an Oracle database on NetApp storage.

This document addresses the requirements for operating an Oracle database on NetApp storage in two ways. First, when a clear best practice exists, it is called out specifically. Second, this document reviews the many design considerations that must be addressed by architects of Oracle storage solutions based on their specific business requirements.

This document first discusses general considerations for all environments followed by specific recommendations based on the choice of virtualization and OS. Special topics such as the choice of file system layout and NFS lock breaking are included in the appendixes.

For more details, see the following additional resources:

- TR-4591: Database Data Protection
- TR-4592: Oracle on MetroCluster
- TR-4534: Migration of Oracle Databases to NetApp Storage Systems

2 ONTAP Platforms

ONTAP software is the foundation for advanced data protection and management. However, ONTAP only refers to software. There are several ONTAP hardware platforms to choose from:

- ONTAP on All Flash FAS (AFF) and FAS
- NetApp Private Storage (NPS) for Cloud
- ONTAP Select
- ONTAP Cloud

The key concept is that ONTAP is ONTAP. Some hardware options offer better performance, others offer lower costs, and some run within hyperscaler clouds. The core functions of ONTAP are unchanged, with multiple replication options available to bind different ONTAP platforms into a single solution. As a result, data protection and disaster recovery strategies can be built on real-world needs, such as performance requirements, capex/opex considerations, and overall cloud strategy. The underlying storage technology runs anywhere in any environment.

2.1 ONTAP with AFF and FAS Controllers

For maximum performance and control of data, ONTAP on a physical AFF or FAS controller remains the leading solution. This is the standard option that thousands of customers have relied upon for more than 20 years. ONTAP delivers solutions for any environment, ranging from three mission-critical databases to 60,000-database service provider deployments, instant restores of petabyte-scale databases, and DBaaS involving hundreds of clones of a single database.

2.2 NPS for Cloud

NetApp introduced the NPS option to address the needs of data-intensive workloads in the public cloud. Although many public cloud storage options exist, most of them have limitations in terms of performance, control, or scale. With respect to database workloads, the primary limitations are as follows:

- Many public cloud storage options do not scale to the IOPS levels required by modern database workloads in terms of cost, efficiency, or manageability.
- Even when the raw IOPS capabilities of a public cloud provider meet requirements, the I/O latencies are frequently unacceptable for database workloads. This has become even more true as databases have migrated to all-flash storage arrays, and businesses have begun to measure latency in terms of microseconds, not milliseconds.
- Although public cloud storage availability is good overall, it does not yet meet the demands of most mission-critical environments.
- Backup and recovery capabilities exist within public cloud storage services, but they generally cannot meet the zero RPO and near-zero RTO requirements of most databases. Data protection requires true instant snapshot-based backup and recovery, not streaming backup and recovery to and from elsewhere in a cloud.
- Hybrid cloud environments must move data between on-premises and cloud storage systems, mandating a common foundation for storage management.
- Many governments have strict data sovereignty laws that prohibit relocating data outside national borders.

NPS systems deliver maximum storage performance, control, and flexibility to public cloud providers, including Amazon AWS, Microsoft Azure, and IBM SoftLayer. This capability is delivered by AFF and FAS systems, including MetroCluster options, in data centers connected directly to public clouds. The full power of the hyperscaler compute layer can be used without the limitations of hyperscaler storage. Furthermore, NPS enables cloud-independent and multicloud architectures because the data, such as application binaries, databases, database backups, and archives, all remain wholly within the NPS system. There is no need to expend time, bandwidth, or money moving data between cloud providers.

Notably, some NetApp customers have used the NPS model on their own initiative. In many locations, high-speed access to one of the hyperscaler providers is readily available to customer data center facilities. In other cases, customers use a colocation facility that is already capable of providing high-speed access to hyperscaler cloud providers. This had led to the use of Amazon AWS, Azure, and SoftLayer as essentially on-demand, consumption-based sources of virtualized servers. In some cases, nothing has changed about the customers' day-to-day operations. They simply use the hyperscaler services as a more powerful, flexible, and cost-efficient replacement for their traditional virtualization infrastructure.

Options are also available for NPS as a service (NPSaaS). In many cases, the demands of database environments are substantial enough to warrant purchasing an NPS system at a colocation facility. However, in some cases, customers prefer to utilize both cloud servers and cloud storage as an operational expense rather than a capital expense. In these cases, they want to use storage resources purely as an as-needed, on-demand service. Several providers now offer NPS as a service for such customers.

2.3 ONTAP Select

ONTAP Select runs on a customer’s own virtualization infrastructure and delivers ONTAP intelligence and data fabric connectivity to the drives inside of white box hardware. ONTAP Select allows ONTAP and guest operating systems to share the same physical hardware as a highly-converged infrastructure. The best practices for running Oracle on ONTAP are not affected. The primary consideration is performance, but ONTAP Select should not be underestimated.

An ONTAP Select environment does not match the peak performance of a high-end AFF system, but most databases do not require 300K IOPS. Typical databases only require around 5K to 10K IOPS, a target that can be met by ONTAP Select. Furthermore, most databases are limited more by storage latency than storage IOPS, a problem that can be addressed by deploying ONTAP Select on SSD drives.

2.4 ONTAP Cloud

ONTAP Cloud is similar to ONTAP Select, except that it runs in a hyperscaler Cloud environment, bringing intelligence and data fabric connectivity to hyperscaler storage volumes. The best practices for running Oracle on ONTAP are not affected. The primary considerations are performance and to a lesser extent cost.

ONTAP Cloud is partially limited by the performance of the underlying volumes managed by the cloud provider. The result is more manageable storage, and, in some cases, the caching capability of ONTAP Cloud offers a performance improvement. However, there are always some limitations in terms of IOPS and latency due to the reliance on the public cloud provider. This does not mean that database performance is unacceptable. It simply means that the performance ceiling is lower than options such as an actual physical AFF system. Furthermore, the performance of storage volumes offered by the various cloud providers that are utilized by ONTAP Cloud are continuously improving.

The prime use case for ONTAP Cloud is currently for development and testing work, but some customers have used ONTAP Cloud for production activity as well. One particularly notable report was the use of Oracle's In-Memory feature to mitigate storage performance limitations. This allows more data to be stored in RAM on the virtual machine hosting the database server, thus reducing performance demands on storage.

3 ONTAP Configuration

A complete description of the configuration of the ONTAP OS is beyond the scope of this document. A best practice for an environment with 2,000 virtualized databases might be inappropriate for a configuration of three very large enterprise resource planning databases. Even small changes in data protection requirements can significantly affect storage design. Some basic details are reviewed in this section. A more complete explanation can be found in TR-4591. For comprehensive assistance with design, contact NetApp or a NetApp partner.

3.1 RAID Levels

Questions occasionally arise concerning RAID levels in the configuration of NetApp storage. Many older Oracle documents and books on Oracle configuration contain warnings about using RAID mirroring and/or avoiding certain types of RAID. Although they raise valid points, these materials do not apply to RAID 4 and the NetApp RAID DP® and RAID-TEC™ technologies used in ONTAP.

RAID 4, RAID 5, RAID 6, RAID DP, and RAID-TEC all leverage parity so that data is not lost because of a drive failure. These RAID options offer much better storage efficiency in comparison to mirroring, but most RAID implementations have a drawback that affects write operations. Completion of a write operation on other RAID implementations requires multiple disk reads to regenerate the parity data, a process commonly called the RAID penalty.

Leveraging ONTAP, however, does not incur a RAID penalty. This is because of the integration of NetApp WAFL® (Write Anywhere File Layout) with the RAID layer. Write operations are coalesced in RAM and prepared as a complete RAID stripe, including parity generation. There is no need to perform a read in order to complete a write, which means that ONTAP and WAFL avoid the RAID penalty. Performance for latency-critical operations, such as redo logging, is unimpeded, and random data-file writes do not incur any RAID penalty resulting from a need to regenerate parity.

With respect to statistical reliability, even RAID DP offers better protection than RAID mirroring. The primary problem is the demand made on disks during a RAID rebuild. With a mirrored RAID set, the risk of data loss from a disk failing while rebuilding to its partner in the RAID set is much greater than the risk of a triple-disk failure in a RAID DP set.

3.2 Capacity Limits

In order to provide high and predictable performance on a storage array, some free space is required for metadata and data organizational tasks. Free space is defined as any space that is not used for actual data, and includes unallocated space on the aggregate itself and unused space within the constituent volumes. Thin provisioning must also be considered. For example, a volume might contain a 1TB LUN of which only 50% is utilized by actual data. In a thin provisioned environment, this would correctly appear to be consuming 500GB of space. However, in a fully provisioned environment, the full capacity of 1TB will appear to be in use. The 500GB of unallocated space will be hidden. This space is unused by actual data and should therefore be included in the calculation of total free space.

NetApp recommendations for storage systems used for databases are described in the sections that follow.

SSD Aggregates, Including AFF Systems

NetApp recommends a minimum of 10% free space. This includes all unused space, including free space within the aggregate or a volume and any free space that is allocated due to the use of full provisioning but is not used by actual data.

The recommendation of 10% free space is conservative. SSD aggregates can support database workloads at even higher levels of utilization without any effect on performance. However, as the utilization of the aggregate increases the risk of running out of space also increases if utilization is not monitored carefully.

HDD Aggregates, Including Flash Pool Aggregates

NetApp recommends a minimum of 15% free space. This includes all unused space, including free space within the aggregate or a volume and any free space that is allocated due to the use of full provisioning but is not used by actual data.

There should be no measurable performance effect when utilization is less than 85%. As utilization approaches 90%, some reduction in performance might become noticeable for certain workloads. As utilization reaches 95%, most database workloads experience a degradation in performance.

3.3 Snapshot-Based Backups

The most important consideration for a file system layout is the plan for leveraging NetApp Snapshot technology. There are two primary approaches:

- Crash-consistent backups
- Snapshot-protected hot backups

A crash-consistent backup of a database requires the capture of the entire database structure, including datafiles, redo logs, and control files, at a single point in time. If the database is stored in a single NetApp FlexVol® flexible volume, then the process is simple; a Snapshot can be created at any time. If a database spans volumes, a consistency group (CG) Snapshot copy must be created. Several options exist for creating CG Snapshot copies, including NetApp SnapCenter® software, the NetApp Snap Creator® framework, NetApp SnapManager® for Oracle (SMO), NetApp SnapDrive® for UNIX, and user-maintained scripts.

Crash-consistent Snapshot backups are primarily used when point-of-the-backup recovery is sufficient. Archive logs can be applied under some circumstances, but when more granular point-in-time recovery is required, a hot backup is preferable.

The basic procedure for a snapshot-based hot backup is as follows:

1. Place the database in `backup` mode.
2. Create a Snapshot copy of all volumes hosting datafiles.
3. Exit `backup` mode.
4. Run the command `alter system archive log current` to force log archiving.
5. Create Snapshot copies of all volumes hosting the archive logs.

This procedure yields a set of Snapshot copies containing datafiles in backup mode and the critical archive logs generated while in backup mode. These are the two requirements for recovering a database. Files such as control files should also be protected for convenience, but the only absolute requirement is protection for datafiles and archive logs.

Although different customers might have very different strategies, almost all of these strategies are ultimately based on the principles outlined in this section.

3.4 Snapshot-Based Recovery

When designing volume layouts for Oracle databases, the first decision is whether to use volume-based NetApp SnapRestore® (VBSR) technology.

Volume-based SnapRestore allows a volume to be almost instantly reverted to an earlier point in time. Because all of the data on the volume is reverted, VBSR might not be appropriate for all use cases. For example, if an entire database, including datafiles, redo logs, and archive logs, is stored on a single volume and this volume is restored with VBSR, then data is lost because the newer archive log and redo data are discarded.

VBSR is not required for restore. Many databases can be restored by using file-based single-file SnapRestore (SFSR) or by simply copying files from the Snapshot copy back into the active file system.

VBSR is preferred when a database is very large or when it must be recovered as quickly as possible, and the use of VBSR requires isolation of the datafiles. In an NFS environment, the datafiles of a given database must be stored in dedicated volumes that are uncontaminated by any other type of file. In a SAN environment, datafiles must be stored in dedicated LUNs on dedicated FlexVol volumes. If a volume manager is used (including Oracle Automatic Storage Management [ASM]), the disk group must also be dedicated to datafiles.

Isolating datafiles in this manner allows them to be reverted to an earlier state without damaging other file systems.

Enhancements in ONTAP 8.2

A significant enhancement to restore capabilities was added in ONTAP 8.2. Previous versions of ONTAP could only create file-level clones from the active file system. With 8.2, it is now possible to create a file-level clone from a Snapshot copy. As a result, it is now much easier to use a file system layout that includes multiple database file types in a single volume or even multiple databases in a single volume.

Prior to version 8.2, a 10TB database would have likely required the isolation of its datafiles in a dedicated volume to deliver an acceptably fast restore procedure. Foreign files in the volume would have precluded the use of VBSR because those foreign files would be destroyed by the restoration process. Recovery would have been performed by copying data instead. This process can still be very fast when managed by a product such as SnapManager for Oracle (SMO), which can invoke an internal copy operation within the array. However, it is still not as fast as VBSR.

With ONTAP 8.2, files and LUNs can be cloned directly from a Snapshot copy. This is a nearly instantaneous process that preserves space efficiency. Therefore, VBSR is no longer required for rapid recovery of large databases, and multiple databases can share a common volume.

In LUN-based environments, store datafiles in dedicated disk groups and LUNs so that they can be restored as a unit. If other files are stored within a datafile disk group, cloning from a Snapshot copy cannot be used for rapid recovery because doing so destroys these foreign files.

Note: When a file is restored by cloning from a Snapshot copy, a background process updates any metadata. There should be no effect on performance, but this process blocks the creation of Snapshot copies until it is complete. The processing rate is approximately 5GBps (18TB/hour) based on the total size of the files restored.

3.5 Snapshot Reserve

For each volume with Oracle data in a SAN environment, the `percent-snapshot-space` should be set to zero because reserving space for a Snapshot copy in a LUN environment is not useful. If the fractional reserve is set to 100, a Snapshot copy of a volume with LUNs requires enough free space in the volume, excluding the snapshot reserve, to absorb 100% turnover of all of the data. If the fractional reserve is set to a lower value, then a correspondingly smaller amount of free space is required, but it always excludes the Snapshot copy reserve. This means that the Snapshot copy reserve space in a LUN environment is wasted.

In an NFS environment, there are two options:

- Set the `percent-snapshot-space` based on expected Snapshot copy space consumption.
- Set the `percent-snapshot-space` to zero and manage active and Snapshot copy space consumption collectively.

With the first option, `percent-snapshot-space` is set to a nonzero value, typically around 20%. This space is then hidden from the user. This value does not, however, create a limit on utilization. If a database with a 20% reservation experiences 30% turnover, the Snapshot copy space can grow beyond the bounds of the 20% reserve and occupy unreserved space.

The main benefit of setting a reserve to a value such as 20% is to verify that some space is always available for Snapshot copies. For example, a 1TB volume with a 20% reserve would only permit a database administrator (DBA) to store 800GB of data. This configuration guarantees at least 200GB of space for Snapshot copy consumption.

When `percent-snapshot-space` is set to zero, all space in the volume is available to the end user, which delivers better visibility. A DBA must understand that, if he or she sees a 1TB volume that leverages Snapshot copies, this 1TB of space is shared between active data and Snapshot turnover.

There is no clear preference between option one and option two among end users.

3.6 read_realloc

Most write activity to an Oracle datafile consists of random overwrites. As these overwrites occur, the changed data is placed on a new physical location within the storage system. This action has no effect on random I/O, which is typically the most performance-critical I/O type. However, it can affect sequential I/O throughput because the storage system is forced to perform more physical disk I/Os to assemble the response to a multiblock read request and to perform readahead.

On an AFF system, the additional I/Os are not significant, but on an array with spinning media, including Flash Pool aggregates, the additional drive head movement results in increased latency and in turn lowers throughput.

Enabling `read_realloc` on a volume results in real-time optimization of the file system layout. When the data on a WAFL volume is poorly allocated, the bulk of the work required to address the problem is read

activity. After the block reads are complete, writing the data back to disk in a single contiguous RAID stripe is a low-cost operation. The `read_realloc` option enables this process in a way that does not affect overall performance.

For example, if a full table scan is being performed, meaning that a datafile is being read sequentially, and `read_realloc` detects blocks that were suboptimally organized on the disk, then 90% of the work to address the problem is already complete. The blocks are already in RAM on the storage system. Therefore, after servicing the read request from the database server, `read_realloc` performs the next step and writes them back to disk in an optimized format. The next time a full table scan is performed, the data is optimized. In the long term, the use of `read_realloc` creates a constant data cleanup process that optimizes the layout of the datafiles on a disk.

There are two `read_realloc` methods: the general `on` and `space_optimized`. The general setting optimizes block layout for both the live file system and the blocks contained in a Snapshot copy. This can result in increased space consumption when Snapshot copies are present, but with the benefit of improved performance during sequential reads on the live file system, Snapshot copies, and clones. If `space_optimized` is used, the blocks contained within Snapshot copies are not reorganized.

These parameters can be changed at any time, but do not enable `read_realloc` across the entire environment at once because the additional work required could affect performance. Enabling it on one or two datafile-containing volumes per day should be a safe approach.

NetApp recommends the following:

- Set `read_realloc` on volumes containing the datafiles and then monitor the space consumption.

  Enabling this option is unnecessary on volumes containing an archive log, control file, or other Oracle file data, but doing so should not cause problems.
- If the Snapshot copies appear to cause excess space consumption, change the setting to `space_optimized`.
- As stated previously, `read_realloc` is not applicable to AFF systems.

3.7 ONTAP and Third-Party Snapshots

Oracle Doc ID 604683.1 explains the requirements for third-party snapshot support and the multiple options available for backup and restore operations.

The third-party vendor must guarantee that the company's snapshots conform to the following requirements:

- Snapshots must integrate with Oracle's recommended restore and recovery operations.
- Snapshots must be database crash consistent at the point of the snapshot.
- Write ordering is preserved for each file within a snapshot.

ONTAP and NetApp Oracle management products comply with these requirements.

3.8 Cluster Operations—Takeover and Switchover

An understanding storage takeover and switchover function is required to ensure database operations are not disrupted by these operations.

- Under normal conditions, incoming writes to a given controller are synchronously mirrored to its partner. In a NetApp MetroCluster™ environment, writes are also mirrored to a remote controller. Until a write is stored in nonvolatile media in all locations, it is not acknowledged to the host application.
- The media storing the write data is called nonvolatile memory or NVMEM. It is also sometimes referred to nonvolatile random access memory (NVRAM), and it can be thought of as a write cache although it functions as a journal. In a normal operation, the data from NVMEM is not read; it is only

used to protect data in the event of a software or hardware failure. When data is written to disk, the data is transferred from the RAM in the system, not from NVMEM.

- During a takeover operation, one node in a high availability (HA) pair takes over the operations from its partner. A switchover is essentially the same, but it applies to MetroCluster configurations in which a remote node takes over the functions of a local node.

During routine maintenance operations, a storage takeover or switchover operation should be transparent, other than for a potential brief pause in database operations as the network paths change. Networking can be complicated and it is easy to make errors, so NetApp strongly recommends testing takeover and switchover operations thoroughly with a database before putting a storage system into production. Doing so is the only way to be sure that all network paths are configured correctly. In a SAN environment, carefully check the output of the command `sanlun lun show -p` to make sure that all expected primary and secondary paths are available.

Care must be taken when issuing a forced takeover or switchover. Forcing a change to storage configuration with these options means that the state of the controller that owns the disks is disregarded and the alternative node forcibly takes control of the disks. Incorrect forcing of a takeover can result in data loss or corruption. This is because a forced takeover or switchover can discard the contents of NVMEM. After the takeover or switchover is complete, the loss of that data means that the data stored on disk might revert to a slightly older state from the point of view of the database.

A forced takeover with a normal HA pair should rarely be required. In almost all failure scenarios, a node shut downs and informs the partner so that an automatic failover takes place. There are some edge cases, such as a rolling failure in which the interconnect between nodes is lost and then one controller is lost, in which a forced takeover is required. In such a situation, the mirroring between nodes is lost before the controller failure, which means that the surviving controller would have no longer has a copy of the writes in progress. The takeover then needs to be forced, which means that data potentially is lost.

The same logic applies to a MetroCluster switchover. In normal conditions, a switchover is nearly transparent. However, a disaster can result in a loss of connectivity between the surviving site and the disaster site. From the point of view of the surviving site, the problem could be nothing more than an interruption in connectivity between sites, and the original site might still be processing data. If a node cannot verify the state of the primary controller, only a forced switchover is possible.

NetApp recommends taking the following precautions:

- Be very careful to not accidentally force a takeover or a switchover. Normally, forcing should not be required, and forcing the change can cause data loss.
- If a forced takeover or switchover is required, make sure that the database is shut down, dismount all file systems, shut down any ASM instances, and varyoff any logical volume manager (LVM) volume groups.
- In the event of a forced MetroCluster switchover, fence off the failed node from all surviving storage resources. For more information, see the MetroCluster Management and Disaster Recovery Guide for the relevant version of ONTAP.

MetroCluster and Multiple Aggregates

MetroCluster is a synchronous replication technology that switches to asynchronous mode if connectivity is interrupted. This is the most common request from customers, because guaranteed synchronous replication means that interruption in site connectivity leads to a complete stall of database I/O, taking the database out of service.

With MetroCluster, aggregates rapidly resynchronize after connectivity is restored. Unlike other storage technologies, MetroCluster should never require a complete remirroring after site failure. Only delta changes must be shipped.

In databases that span aggregates, there is a small risk that additional data recovery steps would be required in a rolling disaster scenario. Specifically, if (a) connectivity between sites is interrupted, (b) connectivity is restored, (c) the aggregates reach a state in which some are synchronized and some are not, and then (d) the primary site is lost, the result is a surviving site in which the aggregates are not synchronized with one another. If this happens, parts of the database are synchronized with one another and it is not possible to bring up the database without recovery. If a database spans aggregates, NetApp strongly recommends leveraging Snapshot-based backups with one of the many available tools to verify rapid recoverability in this unusual scenario.

NVFAIL

Databases are vulnerable to corruption if a failover or switchover is forced because databases maintain large internal caches. If a forced failover occurs, previously acknowledged changes are effectively discarded. The contents of the storage array effectively jump backward in time, and the state of the database cache no longer reflects the state of the data on disk. The result is data corruption.

Caching can occur at the application or server layer. For example, an Oracle database server caches data within the Oracle system global area (SGA). An operation that resulted in lost data would put the database at risk of corruption because the blocks stored in the SGA might not match the blocks on the array. A less obvious use of caching is at the OS file system layer. Blocks from a mounted NFS file system might be cached in the OS, or a file system based on LUNs can cache data in the OS buffer cache. A failure of NVRAM or a forced takeover in these situations could result in file system corruption.

ONTAP systems protect databases and operating systems from this scenario with NVFAIL and its associated parameters.

4 Storage Virtual Machines and Logical Interfaces

This section provides an overview of key management principles. For more comprehensive documentation, see the ONTAP Network Management Guide for the version of ONTAP in use. As with other aspects of database architecture, the best options for storage virtual machine (SVM; formally known as Vserver) and logical interface (LIF) design depend heavily on scaling requirements and business needs.

Consider the following primary topics when building a LIF strategy:

- **Performance.** Is the network bandwidth sufficient?
- **Resiliency.** Are there any single points of failure in the design?
- **Manageability.** Can the network be scaled nondisruptively?

These topics apply to the end-to-end solution, from the host through the switches to the storage system.

4.1 SVMs

SVMs are the basic functional unit of storage, so it is useful to compare an SVM to a guest on a VMware ESX server. When first installed, ESX has no preconfigured capabilities, such as hosting a guest OS or supporting an end-user application. It is an empty container until a virtual machine (VM) is defined. ONTAP is similar. When first installed, this OS has no data-serving capabilities, and an SVM must be defined. It is the SVM personality that defines the data services.

Some customers operate one primary SVM for most of their day-to-day requirements but then create a small number of SVMs for special needs, including the following situations:

- An SVM for a critical business database managed by a specialist team
- An SVM for a development group to whom complete administrative control has been given so that they can manage their own storage independently

- An SVM for sensitive business data, such as human resources or financial reporting data, for which the administrative team must be limited

In a multi-tenant environment, each tenant's data can be given a dedicated SVM. The recommended limit for SVMs is approximately 125 per cluster node, but in general the LIF maximums are reached before the SVM limit is reached. There is a point in which a multi-tenant environment is best separated based on network segments rather than isolating them into dedicated SVMs.

4.2 LIF Types

There are multiple LIF types. Official ONTAP documentation provides more complete information on this topic, but from a functional perspective LIFs can be divided into the following groups:

- **Cluster and node management LIFs.** LIFs used to manage the storage cluster.
- **SVM management LIFs.** Interfaces that permit access to an SVM through the ONTAP API (known as ZAPI) for functions such as Snapshot copy creation or volume resizing. Products such as SMO must have access to an SVM management LIF.
- **Data LIFs.** Interfaces that carry FC, iSCSI, NFS, or CIFS data.

  Note: A data LIF used for NFS traffic can also be used for management by changing the firewall policy from `data` to `mgmt` or another policy that allows HTTP, HTTPS, or SSH. This change can simplify network configuration by avoiding the configuration of each host for access to both the NFS data LIF and a separate management LIF. It is not possible to configure an interface for both iSCSI and management traffic, despite the fact that both use an IP protocol. A separate management LIF is required in iSCSI environments.

4.3 SAN LIF Design

LIF design in a SAN environment is relatively simple for one reason: multipathing. All modern SAN implementations allow a client to access data over multiple network paths and select the best path or paths for access. As a result, performance with respect to LIF design is simpler to address because SAN clients automatically load-balance I/O across the best available paths.

If a path becomes unavailable, the client automatically selects a different path. The resulting simplicity of design makes SAN LIFs generally more manageable. This does not mean that a SAN environment is always more easily managed, because there are many other aspects of SAN storage that are much more complicated than NFS. It simply means that SAN LIF design is easier.

Performance

The most important consideration with LIF performance in a SAN environment is bandwidth. For example, a four-node ONTAP cluster with two 16Gb FC ports per node allows up to 32Gb of bandwidth from each node. I/O is automatically balanced between ports, and all I/O is directed down the most optimal path.

Resiliency

SAN LIFs do not fail over. If a SAN LIF fails, then the client's multipathing ability detects the loss of a path and redirects I/O to a different LIF.

Manageability

LIF migration is a much more common task in an NFS environment because LIF migration is often associated with relocating volumes around the cluster. There is no need to migrate a LIF in a SAN environment when volumes are relocated. That is because, after the volume move has completed, ONTAP sends a notification to the SAN about a change in paths, and the SAN clients automatically reoptimize. LIF migration with SAN is primarily associated with major physical hardware changes. For

example, if a nondisruptive upgrade of the controllers is required, a SAN LIF is migrated to the new hardware. If an FC port is found to be faulty, a LIF can be migrated to an unused port.

Design Recommendations

NetApp makes the following primary recommendations:

- Do not create more paths than are required. Excessive numbers of paths make overall management more complicated and can cause problems with path failover on some hosts. Furthermore, some hosts have unexpected path limitations for configurations such as SAN booting.
- Very few LUNs should require more than four paths to storage. The value of having more than two nodes advertising paths to LUNs is limited because the aggregate hosting a LUN is inaccessible if the node that owns the LUN and its HA partner fail. Creating paths on nodes other than the primary HA pair is not helpful in such a situation.
- Although the number of visible LUN paths can be managed by selecting which ports are included in FC zones, it is generally easier to include all potential target points in the FC zone and control LUN visibility at the ONTAP level.
- In ONTAP 8.3 and later, the selective LUN mapping (SLM) feature is the default. With SLM, any new LUN is automatically advertised from the node that owns the underlying aggregate and the node's HA partner. This arrangement avoids the need to create port sets or configure zoning to limit port accessibility. Each LUN is available on the minimum number of nodes required for both optimal performance and resiliency.

  In the event a LUN must be migrated outside of the two controllers, the additional nodes can be added with the `lun mapping add-reporting-nodes` command so that the LUNs are advertised on the new nodes. Doing so creates additional SAN paths to the LUNs for LUN migration. However, the host must perform a discovery operation to use the new paths.
- Do not be overly concerned about indirect traffic. It is best to avoid indirect traffic in a very I/O-intensive environment for which every microsecond of latency is critical, but the visible performance effect is negligible for typical workloads.
- Follow the zoning rules described in section 13.1.

4.4 NFS LIF Design

In contrast to SAN protocols, NFS has a limited ability to define multiple paths to data. The parallel NFS (pNFS) extensions to NFSv4.1 address this limitation, but pNFS is not yet supported for Oracle databases and is not covered in this document.

Performance and Resiliency

Although measuring SAN LIF performance is primarily a matter of calculating the total bandwidth from all primary paths, determining NFS LIF performance requires taking a closer look at the exact network configuration. For example, two 10Gb ports can be configured as raw physical ports, or they can be configured as a Link Aggregation Control Protocol (LACP) interface group. If they are configured as an interface group, multiple load balancing policies are available that work differently depending on whether traffic is switched or routed. Finally, Direct NFS (DNFS) offers load-balancing configurations that do not exist in any OS NFS clients at this time.

Unlike SAN protocols, NFS file systems require resiliency at the protocol layer. For example, a LUN is always configured with multipathing enabled, meaning that multiple redundant channels are available to the storage system, each of which uses the FC protocol. An NFS file system, on the other hand, depends on the availability of a single TCP/IP channel that can only be protected at the physical layer. This arrangement is why options such as port failover and LACP port aggregation exist.

In an NFS environment, both performance and resiliency are provided at the network protocol layer. As a result, both topics are intertwined and must be discussed together.

Bind LIFs to Port Groups

To bind a LIF to a port group, associate the LIF IP address with a group of physical ports. The primary method for aggregating physical ports together is LACP. The fault-tolerance capability of LACP is fairly simple; each port in an LACP group is monitored and is removed from the port group in the event of a malfunction. There are, however, many misconceptions about how LACP works with respect to performance:

- LACP does not require the configuration on the switch to match the endpoint. For example, ONTAP can be configured with IP-based load balancing, while a switch can use MAC-based load balancing.
- Each endpoint using an LACP connection can independently choose the packet transmission port, but it cannot choose the port used for receipt. This means that traffic from ONTAP to a particular destination is tied to a particular port, and the return traffic could arrive on a different interface. This does not cause problems, however.
- LACP does not evenly distribute traffic all the time. In a large environment with many NFS clients, the result is typically even use of all ports in an LACP aggregation. However, any one NFS file system in the environment is limited to the bandwidth of only one port, not the entire aggregation.
- Although robin-robin LACP policies are available on ONTAP, these policies do not address the connection from a switch to a host. For example, a configuration with a four-port LACP trunk on a host and a four-port LACP trunk on ONTAP is still only able to read a file system using a single port. Although ONTAP can transmit data through all four ports, no switch technologies are currently available that send from the switch to the host through all four ports. Only one is used.

The most common approach in larger environments consisting of many database hosts is to build an LACP aggregate of an appropriate number of 10Gb interfaces by using IP load balancing. This approach enables ONTAP to deliver even use of all ports, as long as enough clients exist. Load balancing breaks down when there are fewer clients in the configuration because LACP trunking does not dynamically redistribute load.

When a connection is established, traffic in a particular direction is placed on only one port. For example, a database performing a full table scan against an NFS file system connected through a four-port LACP trunk reads data though only one network interface card (NIC). If only three database servers are in such an environment, it is possible that all three are reading from the same port, while the other three ports are idle.

Bind LIFs to Physical Ports

Binding a LIF to a physical port results in more granular control over network configuration because a given IP address on a ONTAP system is associated with only one network port at a time. Resiliency is then accomplished through the configuration of failover groups and failover policies.

Failover Policies and Failover Groups

The behavior of LIFs during network disruption is controlled by failover policies and failover groups. Configuration options have changed with the different versions of ONTAP. Consult the ONTAP Network Management Guide for specific details for the version of ONTAP being deployed.

Follow these general practices for ONTAP 8.2 and earlier:

1. Configure a failover group to be user defined.
2. Populate the failover group with ports on the storage failover (SFO) partner controller so that the LIFs follow the aggregates during a storage failover. This configuration avoids the creation of indirect traffic.
3. Use failover ports with performance characteristics that match the original LIF. For example, a LIF on a single physical 10Gb port should include a failover group with a single 10Gb port. A four-port LACP LIF should fail over to another four-port LACP LIF.

4. Set the failover policy to priority.

ONTAP 8.3 allows management of LIF failover based on broadcast domains. Therefore, an administrator can define all of the ports that have access to a given subnet and allow ONTAP to select an appropriate failover LIF. This approach can be used by some customers, but it has limitations in a high-speed database storage network environment because of the lack of predictability. For example, an environment can include both 1Gb ports for routine file system access and 10Gb ports for datafile I/O. If both types of ports exist in the same broadcast domain, LIF failover can result in moving datafile I/O from a 10Gb port to a 1Gb port.

NetApp recommends using the ONTAP 8.2 approach that defines which ports can be used for LIF failover. In summary, consider the following practices:

1. Configure a failover group as user-defined.
2. Populate the failover group with ports on the SFO partner controller so that the LIFs follow the aggregates during a storage failover. This avoids creating indirect traffic.
3. User failover ports with matching performance characteristics to the original LIF. For example, a LIF on a single physical 10Gb port should include a failover group with a single 10Gb port. A four-port LACP LIF should fail over to another four-port LACP LIF. These ports would be a subset of the ports defined in the broadcast domain.
4. Set the failover policy to SFO-partner only. Doing so makes sure that the LIF follows the aggregate during failover.

Auto-revert

Set the `auto-revert` parameter as desired. Most customers prefer to set this parameter to `true` to have the LIF revert to its home port. However, in some cases, customers have set this to `false` so that an unexpected failover can be investigated before returning a LIF to its home port.

LIF-to-Volume Ratio

A common misconception is that there must be a 1:1 relationship between volumes and NFS LIFs. Although this configuration is required for moving a volume anywhere in a cluster while never creating additional interconnect traffic, it is categorically not a requirement. Intercluster traffic must be considered, but the mere presence of intercluster traffic does not create problems. Many of the published benchmarks created for ONTAP include predominantly indirect I/O.

For example, a database project containing a relatively small number of performance-critical databases that only required a total of 40 volumes might warrant a 1:1 volume to LIF strategy, an arrangement that would require 40 IP addresses. Any volume could then be moved anywhere in the cluster along with the associated LIF, and traffic would always be direct, minimizing every source of latency even at microsecond levels.

As a counter example, a large hosted environment might be more easily managed with a 1:1 relationship between customers and LIFs. Over time, a volume might need to be migrated to a different node, which would cause some indirect traffic. However, the performance effect should be undetectable unless the network ports on the interconnect switch are saturating. If there is concern, a new LIF can be established on additional nodes and the host can be updated at the next maintenance window to remove indirect traffic from the configuration.

5 Quality of Service

The increased adoption of all-flash storage has also resulted in consolidation of database workloads. Storage arrays relying on spinning media tended to support only a limited number of databases because of the limited IOPS capabilities of older drive technology. One or two highly active databases would saturate the underlying disks long before the storage controllers reached their limits. This has changed. A

performance capability of a relatively small number of SSD drives can saturate even the most powerful storage controllers. This means the full capabilities of the controllers can be leveraged without the fear of sudden collapse of performance as spinning media latency spiked.

As a reference example, a simple two-node HA AFF8080 system is capable of servicing around 400K random IOPS before latency climbs above one millisecond. Less than one percent of databases would be expected to reach such levels, and allowing an AFF8080 to manage just 10K IOPS would be wasteful.

There are two types of quality of service (QoS) in ONTAP: IOPS and bandwidth. QoS controls can be applied to SVMs, volumes, LUNs, and files.

5.1 IOPS QoS

An IOPS QoS control is obviously based on the total IOPS of a given resource, but there are a number of aspects of IOPS QoS that might not be intuitive. A few customers have been initially puzzled by the apparent increase in latency when an IOPS threshold is reached. This is the only viable method to limit IOPS. Logically, it functions similar to a token system. For example, if a given volume containing datafiles has a 10K IOPS limit, each IO that arrives must first receive a token to continue processing. So long as no more than 10K tokens have been consumed in a given second, no delays are present.

5.2 Bandwidth QoS

First, not all I/O sizes are the same. Databases might be performing a large number of fully random block reads which would result in the IOPS threshold being reached, but databases might also be performing a full table scan operation which would consist of a very small number of large block reads, consuming a very large amount of bandwidth but relatively few IOPS.

5.3 Guaranteed QoS

Many customers seek a solution that includes guaranteed QoS, which is more difficult than it seems and potentially wasteful. For example, placing 10 databases with a 10K IOPS guarantee would require a system to be sized for a scenario where all 10 databases are simultaneously running at 10K IOPS, for a total of 100K. This is a highly unlikely scenario, but if it is required, the best option is to guarantee performance through the sizing effort

For example, each AFF8080 HA pair could constitute a 400K IOPS building block. The total guaranteed IOPS of all hosted databases should not add up to more than 400K IOPS. Furthermore, to prevent a given database from consuming more than its allotted IOPS capability it should have the standard QoS applied as a limit.

6 Compression, Compaction, and Deduplication

Compression and deduplication are two storage efficiency options that increase the amount of logical data that fits on a given amount of physical storage. At a high level, compression is a mathematical process whereby patterns in data are detected and encoded in a way that reduces space requirements. In contrast, deduplication detects actual repeated blocks of data and removes the extraneous copies. Although they deliver similar results, they work in significantly different ways and therefore must be managed differently.

6.1 Compression

There are multiple ways to compress a database. Until recently, compression was of limited value because most databases required a very large number of spindles to provide sufficient performance. One side effect of building a storage array with acceptable performance was that the array generally offered more capacity than required. The situation has changed with the rise of solid-state storage. There is no longer a need to vastly overprovision the drive count to obtain good performance.

Even without compression, migrating a database to a partially or fully solid-state storage platform can yield significant cost savings because doing so avoids the need to purchase drives only needed to support I/O. For example, NetApp has examined some storage configurations from recent large database projects and compared the costs with and without the use of NetApp Flash Cache™ or Flash Pool™ intelligent data caching. These flash technologies decreased costs by approximately 50% because IOPS-dense flash media permits a significant reduction in the number of spinning disks and shelves than would otherwise be required.

As stated above, the increased IOPS capability of solid-state drives (SSDs) almost always yields cost savings, but compression can achieve further savings by increasing the effective capacity of solid-state media. Although compression can be performed by the database itself, this is rarely observed in an Oracle environment. The built-in compression option is not suitable for rapidly changing data, and the advanced compression option has a high licensing cost. In addition, the cost of the Oracle database itself is high. It makes little sense to pay a high per-CPU license cost for a CPU that performs data compression and decompression rather than real database work. A better option is to offload the compression work on to the storage system.

The value of compression is not limited to a pure SSD environment. Hybrid options such as Flash Pool aggregates also benefit from compression. When data is compressed in the SSD layer, the effective result is an increase in SSD storage capacity.

ONTAP 8.3.1

ONTAP 8.3.1 introduces adaptive compression, an inline compression method that works with blocks of varying sizes. The performance effect is minimal, and enabling compression can improve overall performance in some cases. In addition, the compression method available in ONTAP 8.3 and earlier is still available and is now renamed secondary compression.

There is no single best practice for the use of compression; the best option depends on business practices. Most databases can be placed on volumes with adaptive compression enabled with no requirement for separation of data or special treatment of files. As the name implies, adaptive compression adapts to multiple data types and I/O patterns.

Adaptive Compression

Adaptive compression has been thoroughly tested with Oracle workloads, and the performance effect has been found to be negligible, even in an all-flash environment in which latency is measured in microseconds. In initial testing, some customers have reported a performance increase with the use of compression. This increase is the result of compression effectively increasing the amount of Flash Pool SSD available to the database.

ONTAP manages physical blocks in 4KB units. Therefore, the maximum possible compression ratio is 2:1 with a typical Oracle database using an 8KB block. Early testing with real customer data has shown compression ratios approaching this level, but results vary based on the type of data stored.

Secondary Compression

Secondary compression uses a larger block size that is fixed at 32KB. This feature enables ONTAP to compress data with increased efficiency, but secondary compression is primarily designed for data at rest or data that is written sequentially and requires maximum compression.

NetApp recommends secondary compression for data such as archive logs or Recovery Manager (RMAN) backups. These types of files are written sequentially and not updated. This point does not mean that adaptive compression is discouraged. However, if the volume of data being stored is large, then secondary compression delivers better savings when compared to adaptive compression.

Consider secondary compression of datafiles when the amount of data is very large and the datafiles themselves are either read-only or rarely updated. Datafiles using a 32KB block size should see more

compression under secondary compression that has a matching 32KB block size. However, care must be taken to verify that data using block sizes other than 32KB are not placed on these volumes. Only use this method in cases in which the data is not frequently updated.

Caution

Secondary compression and deduplication should not be used together with RMAN backups. The reason is small changes to the backed-up data will affect the 32KB compression window. If the window shifts, the resulting compressed data will differ across the entire file. Deduplication occurs after compression, which means the deduplication engine will see each compressed backup differently. If deduplication of RMAN backups is required, secondary compression should not be used. Adaptive compression is preferable, because it works at a smaller block size and will not disrupt deduplication efficiency. For similar reasons host-side compression will also interfere with deduplication efficiency.

Compression and Thin Provisioning

Compression is a form of thin provisioning. For example, a 100GB LUN occupying a 100GB volume might compress down to 50GB. There are no actual savings realized yet because the volume is still 100GB. The volume must first be reduced in size so the space saved can be used elsewhere on the system. If later changes to the 100GB LUN result in the data becoming less compressible, then the LUN grows in size and the volume could fill up. Thin provisioning can yield a substantial improvement in usable capacity with associated cost savings, but space utilization must be monitored to make sure that capacity is not unexpectedly exhausted.

Alignment

Adaptive compression in a database environment requires some consideration of compression block alignment. Doing so is only a concern for data that is subject to random overwrites of very specific blocks. This approach is similar in concept to overall file system alignment, as is described in the section “Zoning.”

For example, an Oracle 8KB write to a datafile is compressed only if it aligns with an 8KB boundary within the file system itself. This point means that it must fall within the first 8KB of the file, the second 8KB of the file, and so forth. Data such as RMAN backups or archive logs are sequentially written operations that span multiple blocks, all of which are compressed. There is no need to consider alignment. The only I/O pattern of concern is the random overwrites of datafiles.

NFS

With the use of NFS, datafiles are aligned. Each block of the datafile is aligned with respect to the start of the file.

SAN

SAN environments require data to be aligned to an 8KB boundary for optimum compression. There are two aspects of alignment for SAN: the LUN and the file system. The LUN must be configured as either a whole-disk device (no partition) or with a partition that aligns to an 8K boundary. See the OS-specific sections that follow for details on compression and alignment on a configuration-by-configuration basis.

Recommendations for ONTAP 8.3.1 and Later

NetApp provides the following recommendations for ONTAP 8.3.1 and later:

- The simplest approach for leveraging compression is enabling adaptive compression for all database volumes. As stated previously, adaptive compression is suitable for all I/O patterns. Note the following exceptions:

  - If a volume is not thin provisioned, do not enable compression because doing so provides no benefit.
  - If a very large number of archive logs are retained, moving the archive logs to a volume using secondary compression improves storage efficiency.
  - Some databases have very high redo logging rates. Redo logs are comparatively small and are constantly overwritten, so any space savings from compression is negligible. This data should be moved to a volume without compression.
  - If datafiles contain a significant amount of uncompressible data, for example when compression is already enabled or encryption is used, place this data on volumes without compression.
- The exceptions above should not be overemphasized. ONTAP offers the flexibility to choose when compression is enabled, and these exceptions are listed in the interest of offering the broadest choices for customers. Although in most tests the effect of compression is not detectable, it is not nonzero. Some customers seek the highest possible platform performance and minimize every microsecond of latency. In such cases, compression might not be the best option.

Recommendations for ONTAP 8.3 and Earlier

Use ONTAP compression with care in ONTAP 8.3 and earlier because the compression block size is 32K and a typical database uses an 8KB block. As a result, updating a single 8KB block requires ONTAP to read four Oracle blocks, update a single 8KB unit, and write it back to disk.

Some customers have successfully used ONTAP compression in 8.3 and earlier with data that is either written sequentially, such as archived logs, or is not frequently updated, such as archival data in a datafile.

- Do not enable compression on any volumes containing Oracle data unless (a) the change rate is extremely low or (b) the data is written sequentially and not subject to updates, such as for RMAN backup files or archive logs.

ONTAP Version-Specific Notes

The following notes are specific to the ONTAP version in use:

- In ONTAP 8.2, the use of compression on a volume prevents data from being cached by Flash Pool.
- In ONTAP 8.3, compressed blocks are eligible for Flash Pool read caching but not write caching.
- In ONTAP 8.3.1, compressed blocks are eligible for both Flash Pool read and write caching.
- Flash Cache can be used with compressed volumes, but the data is stored in the flash layer in an uncompressed format.

  Note: See the section “Fractional Reservations” for an explanation of the interaction between compression and fractional reservation.

6.2 Inline Data Compaction

Inline data compaction is a technology introduced in ONTAP 9, which improves compression efficiency. As stated previously, adaptive compression alone can provide at best 2:1 savings because it is limited to storing an 8K IO in a 4K WAFL block. Compression methods such as Secondary Compression use a larger block size and deliver better efficiency but are not suitable for data that is subject to small block overwrites. Decompressing 32KB units of data, updating an 8K portion, recompressing, and writing back to disk creates overhead.

Inline data compaction works by allowing logical WAFL blocks to be stored within physical WAFL blocks. For example, a database with highly compressible data such as text or partially full blocks might compress from 8KB to 1KB. Without compaction, that 1KB of data would still occupy an entire 4KB block. Inline data compaction allows that 1KB of compressed data to be stored in just 1KB of physical space

alongside other compressed data. It is not a compression technology, it is simply a more efficient way of allocating space on disk and therefore should not create any detectable performance impact.

The degree of savings obtained will vary. Data that is already compressed or encrypted cannot generally be further compressed and therefore such datasets will not benefit from compaction. Newly initialized Oracle datafiles that contain little more than block metadata and zeros compress up to 80:1. This creates an extremely wide range of possibilities. The best way to evaluate potential savings is using the NetApp Space Savings Estimation Tool (SSET) available on NetApp Field Portal or through your NetApp representative.

6.3 Deduplication

Do not use deduplication with Oracle database files primarily because this process is almost entirely ineffective. An Oracle block contains a header that is globally unique to the database and a trailer that is nearly unique. One percent space savings are possible, but at the expense of significant overhead caused by data deduplication.

Space savings of up to 15% in databases with 16k and large block sizes have been observed in a few cases. The initial 4KB of each block contains the globally unique header, and the final 4KB block contains the nearly unique trailer. The intervening blocks are candidates for deduplication, although in practice this is almost entirely attributed to the deduplication of zeroed data.

Many competing arrays claim the ability to deduplicate Oracle databases based on the presumption that a database is copied multiple times. In this respect, NetApp deduplication could also be used, but ONTAP offers a better option: NetApp FlexClone® technology. The end result is the same; multiple copies of an Oracle database that share most of the underlying physical blocks are created. Using FlexClone is much more efficient than taking the time to copy datafiles and then deduplicate them. It is, in effect, nonduplication rather than deduplication, because a duplicate is never created in the first place.

In the unusual case in which multiple copies of the same datafiles exist, deduplication can be used.

NetApp recommends disabling deduplication on any volume containing Oracle datafiles unless the volume is known to contain multiple copies of the same data, such as repeated RMAN backups to a single volume.

7 Thin Provisioning

Thin provisioning refers to configuring more space on a storage system than is technically available. Such configuring comes in many forms and is integral to the many features that ONTAP offers to an Oracle database environment.

Almost any use of Snapshot involves thin provisioning. For example, a typical 10TB database on NetApp storage contains 30 days of Snapshot copies. This arrangement results in approximately 10TB of data visible in the active file system and 300TB dedicated to Snapshot copies. The total 310TB of storage usually resides on approximately 12TB to 15TB of space. The active database consumes 10TB, and the remaining 300TB of data only requires 2TB to 5TB of space because only the changes to the original data are stored.

Cloning is also an example of thin provisioning. One of NetApp's major customers has created 40 clones of an 80TB database for use by development. If all 40 developers overwrote every block in every datafile, over 3.2PB of storage would be required. In practice, turnover is low and the collective space requirement is closer to 40TB, because only the changes are stored on disk.

7.1 Space Management

Some care must be taken with thin provisioning an Oracle environment because data change rates can increase unexpectedly. For example, space consumption due to Snapshot copies can grow rapidly if

tables are reindexed, or a misplaced RMAN backup can write a large amount of data in a very short time. Finally, it can be difficult to recover an Oracle database if a file system runs out of free space during datafile extension.

Fortunately, these risks can be addressed with careful configuration of `volume-autogrow` and `snapshot-autodelete` policies. As their names imply, these options enable a user to create policies that automatically clear space consumed by Snapshot copies or grow a volume to accommodate additional data. Many options are available, and needs vary by customer.

See the “ONTAP Logical Storage Management Guide” for a complete discussion of these features.

7.2 LUN Thin Provisioning

Thin provisioning of active LUNs is of limited use in an Oracle environment because Oracle initializes datafiles to their full size at the time of creation. The efficiency of thin provisioning of active LUNs in a file system environment can be lost over time as deleted and erased data occupy more and more unallocated whitespace in the file system.

There is one exception when logical volume managers (LVM) are used. When an LVM such as Veritas VxVM or Oracle ASM is used, the underlying LUNs are divided into extents that are only used when needed. For example, if a database begins at 2TB in size but could grow to 10TB over time, this database can be placed on 10TB of thin-provisioned LUNs organized in an LVM disk group. It would occupy only 2TB of disk space at the time of creation and would only claim additional space as extents are allocated to accommodate database growth. This process is safe as long as space is monitored.

7.3 Fractional Reservations

Fractional reserve refers to the behavior of a LUN in a volume with respect to space efficiency. When the option `fractional-reserve` is set to 100%, all data in the volume can experience 100% turnover with any data pattern without exhausting space on the volume.

For a Snapshot example, consider a database on a single 250GB LUN in a 1TB volume. Creating a Snapshot copy would immediately result in the reservation of an additional 250GB of space in the volume to guarantee that the volume does not run out of space for any reason. Using fractional reserves is generally wasteful because it is extremely unlikely that every byte in the database volume would need to be overwritten. There is no reason to reserve space for an event that never happens. Still, if a customer cannot monitor space consumption in a storage system and must be certain that space never runs out, 100% fractional reservations would be a requirement to use Snapshot copies.

7.4 Compression and Deduplication

Compression and deduplication are both forms of thin provisioning. For example, a 50TB database footprint might compress to 30TB, resulting in a savings of 20TB. For compression to yield any benefits, some of that 20TB must be used for other data or the storage system must be purchased with less than 50TB. The result is storing more data than is technically available on the storage system. From the database point of view, there is 50TB of data, even though it occupies only 30TB on disk.

There is always a possibility that the compressibility of a database changes, which would result in increased consumption of real space. This increase in consumption means that compression must be managed as with other forms of thin provisioning in terms of monitoring and using `volume-autogrow` and `snapshot-autodelete`.

Compression and deduplication are discussed in further detail in the sections “Compression” and “Deduplication.”

7.5 Compression and Fractional Reservations

Compression is a form of thin provisioning. Fractional reservations affect the use of compression, with one important note; space is reserved in advance of the Snapshot copy creation. Normally fractional reserve is only important if a Snapshot copy exists. If there is no Snapshot copy, fractional reserve is not important. This is not the case with compression. If a LUN is created on a volume with compression, ONTAP preserves space to accommodate a Snapshot copy. This behavior can be confusing during configuration, but it is expected.

As an example, consider a 10GB volume with a 5GB LUN that has been compressed down to 2.5GB with no Snapshot copies. Consider these two scenarios:

- Fractional reserve = 100 results in 7.5GB utilization
- Fractional reserve = 0 results in 2.5GB utilization

The first scenario includes 2.5GB of space consumption for current data and 5GB of space to account for 100% turnover of the source in anticipation of Snapshot copy use. The second scenario reserves no extra space.

Although this situation might seem confusing, it is unlikely to be encountered in practice. Compression implies thin provisioning, and thin provisioning in a LUN environment requires fractional reservations. It is always possible for compressed data to be overwritten by something uncompressible, which means a volume must be thin provisioned for compression to result in any savings.

NetApp recommends the following reserve configurations:

- Set `fractional-reserve` to 0 when basic capacity monitoring is in place along with `volume-autogrow` and `snapshot-autodelete`.
- Set `fractional-reserve` to 100 if there is no monitoring ability or if it is impossible to exhaust space under any circumstance.

8 Performance Optimization and Benchmarking

Accurate testing of database storage performance is an extremely complicated subject. It requires not just an understanding of IOPS and throughput, but also the understanding of the difference between foreground and background I/O operations, the impact of latency upon the database, and numerous OS and network settings that also affect storage performance. In addition, there are nonstorage databases tasks to consider. There is a point where optimizing storage performance yields no useful benefits because storage performance is no longer a limiting factor for performance.

A majority of database customers now select an all-flash array, which creates some additional considerations. As an example, consider performance testing on a two-node AFF8080 system:

- With a 75/25 read/write ratio, two AFF8080 nodes can deliver over 300K random database IOPS before latency even crosses the 1ms mark. This is so far beyond the current performance demands of most databases that it is difficult to predict the expected improvement. Storage would be largely erased as a bottleneck.
- Network bandwidth is an increasingly common source of performance limitations. For example, spinning disk solutions are often bottlenecks for database performance because the I/O latency is very high. When latency limitations are removed by an all-flash array, the barrier frequently shifts to the network. This is especially notable with virtualized environments and blade systems where the true network connectivity is difficult to visualize. This can complicate performance testing if the storage system itself cannot be fully utilized due to bandwidth limitations.
- Comparing the performance of an all-flash array with an array containing spinning disks is generally not possible because of the dramatically improved latency of all-flash arrays. Test results are typically not meaningful.

- Comparing peak IOPS performance with an all-flash array is frequently not a useful test because databases are not limited by storage I/O. For example, assume one array can sustain 500K random IOPS, whereas another can sustain 300K. The difference is irrelevant in the real world if a database is spending 99% of its time on CPU processing. The workloads never utilize the full capabilities of the storage array. In contrast, peak IOPS capabilities might be critical in a consolidation platform in which the storage array is expected to be loaded to its peak capabilities.
- Always consider latency as well as IOPS in any storage test. Many storage arrays in the market make claims of extreme levels of IOPS, but the latency renders those IOPS useless at such levels. The typical target with all-flash arrays is the 1ms mark. A better approach to testing is not to measure the maximum possible IOPS, but to determine how many IOPS a storage array can sustain before average latency is greater than 1ms.

8.1 Oracle Automatic Workload Repository and Benchmarking

The gold standard for Oracle performance comparison is an Oracle Automatic Workload Repository (AWR) report.

There are multiple types of AWR reports. From a storage point of view, a report generated by running the `awrrpt.sql` command is the most comprehensive and valuable because it targets a specific database instance and includes some detailed histograms that break down storage I/O events based on latency.

Comparing two performance arrays ideally involves running the same workload on each array and producing an AWR report that precisely targets the workload. In the case of a very long-running workload, a single AWR report with an elapsed time that encompasses the start and stop time can be used, but it is preferable to break out the AWR data as multiple reports. For example, if a batch job ran from midnight to 6 a.m., create a series of one-hour AWR reports from midnight–1 a.m., 1 a.m.–2 a.m., and so on.

In other cases, a very short query should be optimized. The best option is an AWR report based on an AWR snapshot created when the query begins and a second AWR snapshot created when the query ends. The database server should be otherwise quiet to minimize the background activity that would obscure the activity of the query under analysis.

Note: Where AWR reports are not available, Oracle statspack reports are a good alternative. They contain most of the same I/O statistics as an AWR report.

8.2 Oracle AWR and Troubleshooting

An AWR report is also the most important tool for analyzing a performance problem.

As with benchmarking, performance troubleshooting requires that you precisely measure a particular workload. When possible, provide AWR data when reporting a performance problem to the NetApp support center or when working with a NetApp or partner account team about a new solution.

When providing AWR data, consider the following requirements:

- Run the `awrrpt.sql` command to generate the report. The output can be either text or HTML.
- If Oracle RAC is used, generate AWR reports for each instance in the cluster.
- Target the specific time the problem existed. The maximum acceptable elapsed time of an AWR report is generally one hour. If a problem persists for multiple hours or involves a multihour operation such as a batch job, provide multiple one-hour AWR reports that cover the entire period to be analyzed.
- If possible, adjust the AWR snapshot interval to 15 minutes. This setting allows a more detailed analysis to be performed. This also requires additional executions of `awrrpt.sql` to provide a report for each 15-minute interval.
- If the problem is a very short running query, provide an AWR report based on an AWR snapshot created when the operation begins and a second AWR snapshot created when the operation ends.

The database server should be otherwise quiet to minimize the background activity that would obscure the activity of the operation under analysis.

- If a performance problem is reported at certain times but not others, provide additional AWR data that demonstrates good performance for comparison.

8.3 calibrate_io

The `calibrate_io` command should never be used to test, compare, or benchmark storage systems. As stated in the Oracle documentation, this procedure calibrates the I/O capabilities of storage.

Calibration is not the same as benchmarking. The purpose of this command is to issue I/O to help calibrate database operations and improve their efficiency by optimizing the level of I/O issued to the host. Because the type of I/O performed by the `calibrate_io` operation does not represent actual database user I/O, the results are not predictable and are frequently not even reproducible.

8.4 SLOB2

SLOB2, the Silly Little Oracle Benchmark, has become the preferred tool for evaluating database performance. It was developed by Kevin Closson and is available here. It takes minutes to install and configure, and it uses an actual Oracle database to generate I/O patterns on a user-definable tablespace. It is one of the few testing options available that can saturate an all-flash array with I/O. It is also useful for generating much lower levels of I/O to simulate storage workloads that are low IOPS but latency sensitive.

8.5 Swingbench

Swingbench can be useful for testing database performance, but it is extremely difficult to use Swingbench in a way that stresses storage. NetApp has not seen any tests from Swingbench that yielded enough I/O to be a significant load on any AFF array. In limited cases, the Order Entry Test (OET) can be used to evaluate storage from a latency point of view. This could be useful in situations where a database has a known latency dependency for particular queries. Care must be taken to make sure that the host and network are properly configured to realize the latency potentials of an all-flash array.

8.6 HammerDB

HammerDB is a database testing tool that simulates TPC-C and TPC-H benchmarks, among others. It can take a lot of time to construct a sufficiently large data set to properly execute a test, but it can be an effective tool for evaluating performance for OLTP and data warehouse applications.

8.7 Orion

The Oracle Orion tool was commonly used with Oracle 9, but it has not been maintained to ensure compatibility with changes in various host operation systems. It is rarely used with Oracle 10 or Oracle 11 due to incompatibilities with OS and storage configuration.

Oracle rewrote the tool, and it is installed by default with Oracle 12c. Although this product has been improved and uses many of the same calls that a real Oracle database uses, it does not use precisely the same code path or I/O behavior used by Oracle. For example, most Oracle I/Os are performed synchronously, meaning the database halts until the I/O is complete as the I/O operation completes in the foreground. Simply flooding a storage system with random I/Os is not a reproduction of real Oracle I/O and does not offer a direct method of comparing storage arrays or measuring the effect of configuration changes.

That said, there are some use cases for Orion, such as general measurement of the maximum possible performance of a particular host-network-storage configuration, or to gauge the health of a storage system. With careful testing, usable Orion tests could be devised to compare storage arrays or evaluate

the effect of a configuration change so long as the parameters include consideration of IOPS, throughput, and latency and attempt to faithfully replicate a realistic workload.

9 General Oracle Configuration

The following parameters are generally applicable to all configurations.

9.1 filesystemio_options

The Oracle initialization parameter `filesystemio_options` controls the use of asynchronous and direct I/O. Contrary to common belief, asynchronous and direct I/O are not mutually exclusive. NetApp has observed that this parameter is frequently misconfigured in customer environments, and this misconfiguration is directly responsible for many performance problems.

Asynchronous I/O means that Oracle I/O operations can be parallelized. Before the availability of asynchronous I/O on various OSs, users configured numerous dbwriter processes and changed the server process configuration. With asynchronous I/O, the OS itself performs I/O on behalf of the database software in a highly efficient and parallel manner. This process does not place data at risk, and critical operations, such as Oracle redo logging, are still performed synchronously.

Direct I/O bypasses the OS buffer cache. I/O on a UNIX system ordinarily flows through the OS buffer cache. This is useful for applications that do not maintain an internal cache, but Oracle has its own buffer cache within the SGA. In almost all cases, it is better to enable direct I/O and allocate server RAM to the SGA rather than to rely on the OS buffer cache. The Oracle SGA uses the memory more efficiently. In addition, when I/O flows through the OS buffer, it is subject to additional processing, which increases latencies. The increased latencies are especially noticeable with heavy write I/O when low latency is a critical requirement.

The options for `filesystemio_options` are:

- **async.** Oracle submits I/O requests to the OS for processing. This process allows Oracle to perform other work rather than waiting for I/O completion and thus increases I/O parallelization.
- **directio.** Oracle performs I/O directly against physical files rather than routing I/O through the host OS cache.
- **none.** Oracle uses synchronous and buffered I/O. In this configuration, the choice between shared and dedicated server processes and the number of dbwriters are more important.
- **setall.** Oracle uses both asynchronous and direct I/O.

In almost all cases, the use of `setall` is optimal, but consider the following issues:

- Some customers have encountered asynchronous I/O problems in the past, especially with previous Red Hat Enterprise Linux 4 (RHEL4) releases. These problems are no longer reported, however, and asynchronous I/O is stable on all current OSs.
- If a database has been using buffered I/O, a switch to direct I/O might also warrant a change in the SGA size. Disabling buffered I/O eliminates the performance benefit that the host OS cache provides for the database. Adding RAM back to the SGA repairs this problem. The net result should be an improvement in I/O performance.
- Although it is almost always better to use RAM for the Oracle SGA than for OS buffer caching, it might be impossible to determine the best value. For example, it might be preferable to use buffered I/O with very small SGA sizes on a database server with many intermittently active Oracle instances. This arrangement allows the flexible use of the remaining free RAM on the OS by all running database instances. This is a highly unusual situation, but it has been observed at some customer sites.

Note: The `filesystemio_options` parameter has no effect in DNFS and ASM environments. The use of DNFS or ASM automatically results in the use of both asynchronous and direct I/O.

NetApp recommends the following:

- Set `filesystemio_options` to `setall`, but be aware that under some circumstances the loss of the host buffer cache might require an increase in the Oracle SGA.

9.2 db_file_multiblock_read_count

The `db_file_multiblock_read_count` parameter controls the maximum number of Oracle database blocks that Oracle reads as a single operation during sequential I/O. This parameter does not, however, affect the number of blocks that Oracle reads during any and all read operations, nor does it affect random I/O. Only sequential I/O is affected.

Oracle recommends that the user leave this parameter unset. Doing so allows the database software to automatically set the optimum value. This generally means that this parameter is set to a value that yields an I/O size of 1MB. For example, a 1MB read of 8KB blocks would require 128 blocks to be read, and the default value for this parameter would therefore be 128.

Most database performance problems observed by NetApp at customer sites involve an incorrect setting for this parameter. There were valid reasons to change this value with Oracle versions 8 and 9. As a result, the parameter might be unknowingly present in `init.ora` files because the database was upgraded in place to Oracle 10 and later. A legacy setting of 8 or 16, compared to a default value of 128, significantly damages sequential I/O performance.

NetApp recommends the following:

- The `db_file_multiblock_read_count` parameter should not be present in the `init.ora` file. NetApp has never encountered a situation in which changing this parameter improved performance, but there are many cases in which it caused clear damage to sequential I/O throughput.

9.3 Redo Block Size

Oracle supports either a 512-byte or 4KB redo block size. The default is 512 bytes. The best option is expected to be 512 bytes because this size minimizes the amount of data written during redo operations. However, it is possible that the 4KB size could offer a performance benefit at very high logging rates. For example, a single database with 50MBps of redo logging might be more efficient if the redo block size is larger. A storage system supporting many databases with a large total amount of redo logging might benefit from a 4KB redo block size. This is because this setting would eliminate inefficient partial I/O processing when only a part of a 4KB block must be updated.

It is not correct that all I/O operations are performed in single units of the redo log block size. At very high logging rates, the database generally performs very large I/O operations composed of multiple redo blocks. The actual size of those redo blocks does not generally affect the efficiency of logging.

NetApp recommends the following:

- Only change the default block size for cause, such as a documented requirement for a particular application or because of a recommendation made by NetApp or Oracle customer support.

9.4 Checksums and Data Integrity

One question commonly directed to NetApp is how to secure the data integrity of a database. This question is particularly common when a customer who is accustomed to using Oracle RMAN streaming backups migrates to snapshot-based backups. One feature of RMAN is that it performs integrity checks during backup operations. Although this feature has some value, its primary benefit is for a database that is not used on a modern storage array. When physical disks are used for an Oracle database, it is nearly

certain that corruption eventually occurs as the disks age, a problem that is addressed by array-based checksums in true storage arrays.

With a real storage array, data integrity is protected by using checksums at multiple levels. If data is corrupted in an IP-based network, the Transmission Control Protocol (TCP) layer rejects the packet data and requests retransmission. The FC protocol includes checksums, as does encapsulated SCSI data. After it is on the array, ONTAP has RAID and checksum protection. Corruption can occur, but, as in most enterprise arrays, it is detected and corrected. Typically, an entire drive fails, prompting a RAID rebuild, and database integrity is unaffected. Less often, ONTAP detects a checksum error, meaning that data on the disk is damaged. The disk is then failed out and a RAID rebuild begins. Once again, data integrity is unaffected.

The Oracle datafile and redo log architecture is also designed to deliver the highest possible level of data integrity, even under extreme circumstances. At the most basic level, Oracle blocks include checksum and basic logical checks with almost every I/O. If Oracle has not crashed or taken a tablespace offline, then the data is intact. The degree of data integrity checking is adjustable, and Oracle can also be configured to confirm writes. As a result, almost all crash and failure scenarios can be recovered, and in the extremely rare event of an unrecoverable situation, corruption is promptly detected.

Most NetApp customers using Oracle databases discontinue the use of RMAN and other backup products after migrating to snapshot-based backups. There are still options in which RMAN can be used to perform block-level recovery with SMO. However, on a day-to-day basis, RMAN, NetBackup, and other products are only used occasionally to create monthly or quarterly archival copies.

Some customers choose to run `dbv` periodically to perform integrity checks on their existing databases. NetApp discourages this practice because it creates unnecessary I/O load. As discussed above, if the database was not previously experiencing problems, the chance of `dbv` detecting a problem is close to zero, and this utility creates a very high sequential I/O load on the network and storage system. Unless there is reason to believe corruption exists, such as exposure to a known Oracle bug, there is no reason to run `dbv`.

10 Flash

A comprehensive explanation of the use of flash and SSD technologies with Oracle databases is beyond the scope of this document, but some common questions and misconceptions must be addressed. All principles explained in this section apply equally to all protocols and file systems, including Oracle ASM.

10.1 Flash Cache

NetApp Flash Cache intelligent data caching has been the leading flash-based technology used in Oracle deployments for a simple reason: most databases are limited by random read latency. Flash Cache is a simple method for accelerating random read performance.

However, Flash Cache does have the limitation of being tied to the particular node that hosts the PCIe card containing the flash memory. As spindle sizes increase, customers deploy fewer spindles. This increases the risk that controller failure results in a period of performance degradation while the Flash Cache card on the takeover node warms up. Until warmup is complete, there can be significantly more I/O on the spinning media. For this reason, NetApp Flash Pool intelligent data caching is the preferred flash technology because the flash layer follows the spinning media during takeover. No warm-up time is required, and the cache does not go cold. This does not mean, however, that configurations with Flash Cache are flawed. Most systems contain enough spindles to cope with increased I/O during a controller takeover.

In general, the default settings are optimal for database workloads. Make sure that Flash Cache is enabled by setting `flexscale. enable=on`.

flexscale.lopri_blocks

The `flexscale.lopri_blocks` parameter applies to the use of Flash Cache intelligent caching. The default for this option is `off`, which means that I/O from low-priority block operations such as random overwrites and sequential I/O are not cached. The reason is simple; most databases are limited by latency on random-read operations. When a random overwrite occurs, an Oracle database almost always retains a copy of that block, and the block is highly unlikely to be reread soon. Caching such overwrites wastes valuable space in Flash Cache. When Oracle performs sequential read I/O, it is a very large block operation that is processed with inherent efficiency by the storage array, even if the underlying disk is SATA. This type of I/O does not benefit from Flash Cache. Attempting to cache this I/O generally places unnecessary load on the CPU and again wastes valuable space in Flash Cache that would be better used for caching random I/O.

NetApp recommends the following:

- Only change this parameter after careful consultation with NetApp customer support or professional services or after thorough testing.

flexscale.read-ahead_blocks

The `flexscale.read-head_blocks` parameter was added in ONTAP 8.2, and is similar to `flexscale.lopri_blocks` except that it only targets read data. Under normal operation, Flash Cache only stores randomly accessed data. Enabling `flexscale.read-ahead_blocks` enables caching of sequentially read data. As discussed above, sequential I/O is already very efficient and does not generally benefit from being stored in flash memory.

NetApp recommends the following:

- Only change this parameter after careful consultation with NetApp Customer Support or Professional Services or after thorough testing.

flexscale.random_write_through_blocks

The `flexscale.random_write_through_blocks` parameter was added in ONTAP 8.3. Unlike the prior two options, this parameter has the potential to help Oracle workloads. In most cases, randomly written data does not need to be cached because the database retains a copy of the block. In situations in which the Oracle cache is under pressure, randomly written blocks might be read back again quickly and capturing that data in Flash Cache improves performance.

Refer to the following discussion of Flash Pool write caching for a more complete explanation of the value of write caching.

NetApp recommends the following:

- By default, set this parameter to `off`. However, NetApp recommends experimenting with alternative settings. A database that is limited by `db_file_sequential_read` performance is a candidate for enabling `random_write_through_blocks`. A decrease in latency is observed if the database rapidly rereads recently written blocks.

10.2 SSD Aggregates

There is a lot of confusion about the use of SSD and Flash media for redo logs. Good redo logging performance requires that the data be written to SSD. An SSD drive can be valuable for improving logging performance when used with directly connected devices, but NetApp storage arrays already contain nonvolatile, mirrored, NVRAM-based or NVMEM-based solid-state storage. When an Oracle database performs a write operation, the write is acknowledged as soon as it is journaled into NVRAM or NVMEM. Write performance is not directly affected by the type of drives that eventually receive the writes.

At best, use of an SSD aggregate or AFF platform for hosting sequential writes such as redo logging or for temporary datafile I/O has no effect. There are circumstances where choosing AFF will improve write performance indirectly, though. For example, a system with heavy random I/O that is overloading spinning media might reach a point where the drives are no longer able to absorb the incoming writes quick enough to keep NVMEM/NVRAM from filling up. In these cases, a change to an SSD aggregate or AFF platform can improve redo performance, but it is an indirect benefit. The write performance problem would be resolved because the system is better able to process random IO. The write behavior then returns to normal, with all inbound writes committing to NVMEM/NVRAM without delay.

On occasion, customers have made planning errors which result in performance damage with an SSD aggregate. Although SSD drives offer far higher performance than spinning media, SSD aggregates sometimes have far fewer devices than do SAS or SATA aggregates on the system. For example, NetApp has observed severe performance problems in customer environments caused by moving heavy sequential-write workloads, including redo logs, from a large SAS aggregate that might contain 100 drives to a small SSD aggregate with only 4 or 5 devices. SSD drives might be faster than SAS, but they are not unlimited.

The primary application for an SSD aggregate is servicing random IO workloads. Indexes are particularly good candidates for placement on SSD drives. Other types of IO should not suffer so long as there is not an excessively small number of drives in the aggregate, but a performance improvement should not be expected unless the prior system was badly overloaded.

10.3 Flash Pool

The same principles that underlie Flash Cache and SSD aggregates also apply to Flash Pool intelligent caching. Flash Pool improves the latency of random reads, which is typically the primary performance bottleneck in an Oracle database. Flash Pool is also a cost-saving technology. Many Oracle storage systems have a significant number of spindles that service bursts of random-read activity at minimum latency. A small Flash Pool allocation can replace a large number of spinning drives.

A further benefit of Flash Pool not realized with Flash Cache is write caching or, specifically, overwrite caching. Using Flash Pool for write caching does not directly affect write performance because writes commit to NVRAM or NVMEM first. From a latency perspective, I/O is complete when data is journaled into NVRAM or NVMEM. The type of media on which an inbound write is subsequently stored does not affect performance by itself. There can, however, be an indirect benefit to write performance if the use of Flash Pool write caching decreases pressure on spinning media and thus leads to a general improvement in I/O performance on the entire array.

Flash Pool write caching also improves read latency in which randomly overwritten blocks are rapidly read again. This process is not applicable to all databases because the database typically retains a copy of a written block. As the size of the Oracle buffer cache increases and more writes are cached, it becomes less likely that the block must be reread from disk. In such cases, it might be preferable to disable write caching and reserve valuable flash space for random read operations.

On the other hand, there is a benefit to capturing repeated overwrites of the same blocks on the SSD layer to reduce pressure on spinning media. This issue can occur when the Oracle buffer cache is under pressure and blocks are aging out of cache only to be read again quickly.

NetApp recommends the following:

- Retain the default Flash Pool policy, which includes both random-read and random-write caching.
- Although write caching might not be beneficial, the overall random-write levels observed in most Oracle databases are not high enough to cause excessive use of SSD space. The defaults make write caching available if needed.

The primary exception is a database workload with the following characteristics:

- The workload dominates an aggregate and is therefore responsible for most of the Flash Pool caching activity.
- The workload is known to be limited by random-read latency.
- Write activity is relatively low.
- The Oracle buffer cache is relatively large.

In such cases, changing the Flash Pool write cache policy to `none` might be warranted. Maximum space would then be available on the SSDs for read caching.

Flash Pool is often useful for Oracle standby databases, including use with Oracle DataGuard, because a standby database usually lacks a true buffer cache. This situation results in a demanding I/O pattern in which the same blocks are read, updated, and written repeatedly. Flash Pool captures this concentrated overwrite activity in the SSD layer, which reduces pressure on spinning media. Prior to the availability of technologies such as Flash Pool, it was not unusual for a standby database to require more spinning disks than the primary database that was the source of replication.

10.4 AFF Platforms

NetApp AFF extends the value of SSD aggregates through increased performance and default behavior tuned for an all-flash platform. Complete documentation is available on the NetApp Support site.

One particular consideration that should be made is that flash is not exclusively about IOPS. There are other benefits such as consistency and predictability of performance, decreased power consumption, decreased heat output, and general future-proofing of a solution.

In many cases, an all-flash platform can decrease costs as it avoids the need to deploy drive after drive of spinning media purely to ensure good latency. Costs continue to decrease dramatically, which leads more and more customers to select AFF as the default choice.

11 Ethernet Configuration

The TCP/IP settings required for Oracle database software installation are usually sufficient to provide good performance for all NFS or iSCSI storage resources. In some cases, NetApp has seen performance benefits in 10Gb environments after implementing specific recommendations from the network adapter manufacturer.

11.1 Ethernet Flow Control

This technology allows a client to request that a sender temporarily stop data transmission. This is usually done because the receiver is unable to process incoming data quickly enough. At one time, requesting that a sender cease transmission was less disruptive than having a receiver discard packets because buffers were full. This is no longer the case with the TCP stacks used in OSs today. In fact, flow control causes more problems than it solves.

Performance problems caused by Ethernet flow control have been increasing in recent years. This is because Ethernet flow control operates at the physical layer. If a network configuration permits any database server to send an Ethernet flow control request to a storage system, the result is a pause in I/O for all connected clients. Because an increasing number of clients are served by a single storage controller, the likelihood of one or more of these clients sending flow control requests increases. The problem has been seen frequently at customer sites with extensive OS virtualization.

A NIC on a NetApp system should not receive flow-control requests. The method used to achieve this result varies based on the network switch manufacturer. In most cases, flow control on an Ethernet switch can be set to `receive desired` or `receive on`, which means that a flow control request is not forwarded to the storage controller. In other cases, the network connection on the storage controller might not allow flow-control disabling. In these cases, the clients must be configured to never send flow control

requests, either by changing to the NIC configuration on the database server itself or the switch ports to which the database server is connected.

NetApp recommends the following:

- Make sure that NetApp storage controllers do not receive Ethernet flow-control packets. This can generally be done by setting the switch ports to which the controller is attached, but some switch hardware has limitations that might require client-side changes instead.

11.2 Jumbo Frames

The use of jumbo frames has been shown to offer some performance improvement in 1Gb networks by reducing CPU and network overhead, but the benefit is not usually significant. Even so, NetApp recommends implementing jumbo frames when possible, both to realize any potential performance benefits and to future-proof the solution.

Using jumbo frames in a 10Gb network is almost mandatory. This is because most 10Gb implementations reach a packets-per-second limit without jumbo frames before they reach the 10Gb mark. Using jumbo frames improves efficiency in TCP/IP processing because it allows the database server, NICs, and the storage system to process fewer larger packets. The performance improvement varies from NIC to NIC, but it is significant.

In jumbo-frame implementations, there are common but incorrect beliefs that all connected devices must support jumbo frames and that the MTU size must match end-to-end. Instead, the two network end points negotiate the highest mutually acceptable frame size when establishing a connection. In a typical environment, a network switch is set to an MTU size of 9216, the NetApp controller is set to 9000, and the clients are set to a mix of 9000 and 1514. Clients that can support an MTU of 9000 can use jumbo frames, and clients that can only support 1514 can negotiate a lower value.

Problems with this arrangement are rare in a completely switched environment. However, take care in a routed environment that no intermediate router is forced to fragment jumbo frames.

NetApp recommends the following:

- Jumbo frames are desirable but not required with 1Gb Ethernet (GbE)
- Jumbo frames are required for maximum performance with 10GbE

11.3 TCP Parameters

Three settings are frequently misconfigured: TCP timestamps, selective acknowledgment (SACK), and TCP window scaling. Many out-of-date documents on the Internet recommend disabling one or more of these parameters to improve performance. There was some merit to this recommendation many years ago when CPU capabilities were much lower and there was a benefit to reducing the overhead on TCP processing whenever possible.

However, with modern OSs, disabling any of these TCP features usually results in no detectable benefit or might result in damage performance. Performance damage is especially likely in virtualized networking environments because these features are required for efficient handling of packet loss and changes in network quality.

NetApp recommend the following:

- Enable TCP timestamps, SACK, and TCP window scaling on the host

12 General NFS Configuration

12.1 NFS Versions

Oracle currently limits NFS support to NFS version 3. For this reason, NetApp does not support the use of NFSv4, NFSv4.1, or pNFS with Oracle databases. This document will be updated if Oracle's support stance changes.

NetApp recommends the following:

- NFSv3 is mandatory at this time

12.2 TCP Slot Tables

TCP slot tables are the NFS equivalent of host bus adapter (HBA) queue depth. These tables control the number of NFS operations that can be outstanding at any one time. The default value is usually 16, which is far too low for optimum performance. The opposite problem occurs on newer Linux kernels, which can automatically increase the TCP slot table limit to a level that saturates the NFS server with requests.

For optimum performance and to prevent performance problems, adjust the kernel parameters that control the TCP slot tables.

Run the `sysctl -a | grep tcp.*.slot_table` command, and observe the following parameters:

```
# sysctl -a | grep tcp.*.slot_table
sunrpc.tcp_max_slot_table_entries = 128
sunrpc.tcp_slot_table_entries = 128
```

All Linux systems should include `sunrpc.tcp_slot_table_entries`, but only some will include `sunrpc.tcp_max_slot_table_entries`. They should both be set to 128.

12.3 Installation and Patching

The presence of the following mount options in `ORACLE_HOME` causes host caching to be disabled:

```
cio, actimeo=0, noac, forcedirectio.
```

This action can have a severe negative effect on the speed of Oracle software installation and patching. Many customers temporarily remove these mount options during installation or patching of the Oracle binaries. This removal can be performed safely if the user verifies that no other processes are actively using the target `ORACLE_HOME` during the installation or patching process.

12.4 ONTAP and NFS Flow Control

Under some circumstances, the use of ONTAP requires changes in the Oracle or Linux kernel parameter. The reason is related to NFS flow control, so do not confuse these changes with Ethernet flow control. NFS flow control enables an NFS server such as ONTAP to limit network communication with an NFS client that is not acknowledging receipt of data. This capability protects the NFS server in cases in which a malfunctioning NFS client requests data at a rate beyond its ability to process the responses. Without protection, the network buffers on the NFS server fill up with unacknowledged packets.

Under rare circumstances, I/O bursts from both Oracle DNFS clients and newer Linux NFS clients can exceed the limits at which the ONTAP NFS server can protect itself. The NFS client lags in its processing of inbound data while continuing to send requests for more data. This lag can lead to performance and stability problems with NFS connectivity.

Although problems are rare, NetApp recommends the following protective measures as best practices. These measures apply only to ONTAP, and the changes should not adversely affect performance.

NetApp recommends the following with ONTAP:

- When Oracle DNFS is used, set the `DNFS_BATCH_SIZE` parameter to `128`. This parameter is available with Oracle 11.2.0.4 and later. If this cannot be done, do not use DNFS.
- Make sure that both of the TCP slot tables parameters discussed previously are set to 128.
- The Oracle `calibrate_io` command does not work if `DNFS_BATCH_SIZE` is set to anything other than the default value. If I/O needs to be calibrated, temporarily remove the `DNFS_BATCH_SIZE` parameter during calibration.

12.5 Direct NFS

Oracle's DNFS client is designed to bypass the host NFS client and perform NFS file operations directly on an NFS server. Enabling it only requires changing the Oracle Disk Manager library. Instructions for this process are provided in the Oracle documentation.

Using DNFS results in a general improvement in I/O performance and decreases the load on the host and the storage system because I/O is performed in the most efficient way possible. In addition, Oracle DNFS provides multipathing and fault-tolerance. For example, two 10Gb interfaces can be bound together to offer 20Gb of bandwidth. A failure of one interface results in I/Os being retried on the other interface. The overall operation is very similar to FC multipathing.

When DNFS is used, it is critical that all patches described in Oracle Doc 1495104.1 are installed. If a patch cannot be installed, the environment must be evaluated to make sure that the bugs described in that document do not cause problems. In some cases, an inability to install the required patches prevents the use of DNFS.

Caution
• Before using DNFS, verify that the patches described in Oracle Doc 1495104.1 are installed. • Starting with Oracle 12c, DNFS includes support for NFSv3, NFSv4, and NFSv4.1. NetApp support policies cover v3 and v4 for all clients, but at the time of writing NFSv4.1 is not supported for use with Oracle DNFS. • Do not use DNFS with any type of round-robin name resolution, including DNS, DDNS, NIS or any other method. This includes the DNS load balancing feature available in ONTAP. When an Oracle database using DNFS resolves a host name to an IP address it must not change on subsequent lookups. This can result in Oracle database crashes and possible data corruption.

12.6 Direct NFS and Host File System Access

Using DNFS can occasionally cause problems for applications or user activities that rely on the visible file systems mounted on the host because the DNFS client accesses the file system out of band from the host OS. The DNFS client can create, delete, and modify files without the knowledge of the OS.

When the mount options for single-instance databases are used, they enable caching of file and directory attributes, which also means that the contents of a directory are cached. Therefore, DNFS can create a file, and there is a short lag before the OS rereads the directory contents and the file becomes visible to the user. This is not generally a problem, but, on rare occasions, utilities such as SAP BR*Tools might have issues. If this happens, address the problem by changing the mount options to use the recommendations for Oracle RAC. This change results in the disabling of all host caching.

Only change mount options when (a) DNFS is used and (b) a problem results from a lag in file visibility. If DNFS is not in use, using Oracle Real Application Cluster (RAC) mount options on a single-instance database results in degraded performance.

Note: See the note about `nosharecache` in the section "Linux NFSv3 Mount Options" for a Linux-specific DNFS issue that can produce unusual results.

12.7 ADR and NFS

Some customers have reported performance problems resulting from an excessive amount of I/O on data in the `ADR` location. The problem does not generally occur until a lot of performance data has accumulated. The reason for the excessive I/O is unknown, but this problem appears to be a result of Oracle processes repeatedly scanning the target directory for changes.

Removal of the `noac` and/or `actimeo=0` mount options allows host OS caching to occur and reduces storage I/O levels.

NetApp recommends the following:

- Do not place `ADR` data on a file system with `noac` or `actimeo=0` because performance problems are likely. Separate `ADR` data into a different mount point if necessary.

13 General SAN Configuration

13.1 Zoning

An FC zone should never contain more than one initiator. Such an arrangement might appear to work initially, but crosstalk between initiators eventually interferes with performance and stability.

Multitarget zones are generally regarded as safe, although in rare circumstances the behavior of FC target ports from different vendors has caused problems. For example, avoid including the target ports from both a NetApp and an EMC storage array in the same zone. In addition, placing a NetApp storage system and a tape device in the same zone is even more likely to cause problems.

13.2 LUN Alignment

LUN alignment refers to optimizing I/O with respect to the underlying file system layout. On a NetApp system, storage is organized in 4KB units. Align an 8KB block on an Oracle datafile to exactly two 4KB blocks. If an error in LUN configuration shifts the alignment by 1KB in either direction, each 8KB Oracle block would exist on three different 4KB storage blocks rather than two. This arrangement would cause increased latency and cause additional I/O to be performed within the storage system.

LUN alignment is generally only a concern when a logical volume manager is not used. As a practical matter, this means that Linux and Solaris are of primary concern. If a physical volume within a logical volume group is defined on the whole disk device (no partitions are created), the first 4KB block on the LUN aligns with the first 4KB block on the storage system. This is a correct alignment. Problems arise with partitions because they shift the starting location where the OS uses the LUN. As long as the offset is shifted in whole units of 4KB, the LUN is aligned.

In Linux environments, build logical volume groups on the whole disk device. When a partition is required, check alignment by running `fdisk –u` and verifying that the start of each partition is a multiple of eight. This means that the partition starts at a multiple of eight 512-byte sectors, which is 4KB.

Also see the discussion about compression block alignment in the section “Compression.” Any layout that is aligned with 8KB compression block boundaries is also aligned with 4KB boundaries.

Alignment in Solaris environments is more complicated. Refer to the appropriate Host Utilities documentation for more information.

Caution

In Solaris x86 environments, take additional care about proper alignment because most configurations have several layers of partitions. Solaris x86 partition slices usually exist on top of a standard master boot record partition table.

13.3 LUN Misalignment Warnings

Oracle redo logging normally generates unaligned I/O that can cause misleading warnings about misaligned LUNs on ONTAP. Oracle redo logging performs a sequential overwrite of the redo log file with writes of varying size. A log write operation that does not align to 4KB boundaries does not ordinarily cause performance problems because the next log write operation completes the block. The result is that ONTAP is able to process almost all writes as complete 4KB blocks, even though the data in some 4KB blocks was written in two separate operations.

Verify alignment by using by using utilities such as `sio` or `dd` that can generate I/O at a defined block size. The I/O alignment statistics on the storage system can be viewed with the `stats` command. See "Appendix B: WAFL Alignment Verification" for more information.

13.4 LUN Sizing

A LUN is a virtualized object on ONTAP that exists across all of the spindles in the hosting aggregate. As a result, the performance of the LUN is unaffected by its size because the LUN draws on the full potential of the aggregate no matter which size is chosen.

As a matter of convenience, customers might wish to use a LUN of a particular size. For example, if a database is built on an ASM disk group composed of two LUNS of 1TB each, then that ASM disk group must be grown in increments of 1TB. It might be preferable to build the ASM disk group from eight LUNs of 500GB each so that the disk group can be increased in smaller increments.

The practice of establishing a universal standard LUN size is discouraged, because doing so can complicate manageability. For example, a standard LUN size of 100GB might work well when a database is in the range of 1TB to 2TB, but a database 20TB in size would require 200 LUNs. This means that server reboot times are longer, there are more objects to manage in the various UIs, and products such as SMO must perform discovery on many objects. Using fewer, larger LUNs avoids such problems.

Note:

- The LUN count is more important than the LUN size.
- LUN size is mostly controlled by LUN count requirements.
- Avoid creating more LUNs than required.

13.5 LUN Resizing and LVM-Based Resizing

When a SAN-based file system has reached its capacity limit, there are two options for increasing the space available:

- Increase the size of the LUNs.
- Add a LUN to an existing volume group and grow the contained logical volumes.

Both options are supportable, but increasing a LUN size is generally more difficult and can be risky. Some of the considerations are as follows:

- LUNs created by ONTAP can be increased to approximately 10X of their original size. The limitation is based on the inherent structure of disk geometry. There are sometimes options to increase beyond the 10X mark, but it can require changes to partition tables that require advanced understanding of disk configuration at the host level.
- It is recommended that a database be shut down before attempting a LUN resize, and to create Snapshot copies as a fallback measure. Although this is not a requirement in all cases, there is some risk of disruption and risk of user error when rediscovering the newly enlarged LUNs at the host OS level.
- One exception to LUN resizing complications is Microsoft Windows, which offers a safe and nondisruptive method of increasing LUN sizes using NetApp SnapDrive for Windows.

Although LUN resizing is an option to increase capacity, it is generally better to use a logical volume manager (LVM), including Oracle ASM. One of the principle reasons LVMs exist is to avoid the need for a LUN resize. With an LVM, multiple LUNs are bonded together into a virtual pool of storage. The logical volumes carved out of this disk pool are managed by the LVM and can be easily resized. An additional benefit is the avoidance of hotspots on a particular disk by distributing a given logical volume across all available LUNs. Transparent migration can usually be performed by using the volume manager to relocate the underlying extents of a logical volume to new LUNs.

For the above reasons, a strategy involving resizing LUNs is discouraged in favor of an LVM approach.

13.6 LUN Count

Unlike the LUN size, the LUN count does affect performance. Oracle database performance is affected by the capability to perform parallel I/O through the SCSI layer. As a result, two LUNs offer better performance than a single LUN. Using a logical volume manager, such as Veritas VxVM, Linux LVM2, or Oracle ASM is the simplest method to increase parallelism.

NetApp customers have generally experienced minimal benefit from increasing the number of LUNs beyond eight, although the testing of 100%-SSD environments with very heavy random I/O has demonstrated further improvement up to 64 LUNs. NetApp recommends building a volume group with an extent size that enables the even distribution of I/O. For example, a 1TB volume group composed of 10 100GB LUNs and an extent size of 100MB would yield 10,000 extents in total (1,000 extents per LUN). The resulting I/O on a database placed on this 1TB volume group should be evenly distributed across all 10 LUNs.

Distributing a logical volume across extents is not the same as striping, although the concept is similar.

Logical volume managers break up LUNs into relatively large extents in order to make data management simpler. Larger extents are preferred because they deliver better efficiency of readahead operations. For example, an Oracle ASM extent, also called an Allocation Unit, of 64MB allows storage array readahead to assist transferring a full 64MB of data before ASM moves to the next extent. Smaller allocation units mean readahead must reset more often as read operations move from extent to extent.

The more important random IO should still be evenly distributed across LUNs even with a large extent size unless the database has extremely concentrated IO.

In contrast to distributed extents, true striping should be avoided. Striping was mostly targeting relatively slow spinning disk technology. For example, if an application was known to read in 1MB chunks, a stripe set could be created with eight LUNs with a stripe width of 128KB. The result is a 1MB operation could be executed as eight simultaneous 128KB IOs on each LUN. This is almost never beneficial with a modern database and storage system. Furthermore, incorrect tuning of a striped volume group will result in damage to performance.

Most databases are limited by random I/O performance, not sequential performance. A datafile that exists across a large number of extents enables a large amount of random I/O to be randomized across many extents. This arrangement means that all LUNs in the volume group are used evenly and no individual LUN limits performance.

NetApp recommends the following:

- In general, four to eight LUNs are sufficient to support datafile I/O. Less than four LUNs might create performance limitations because of limitations in host SCSI implementations.
- Do not use fine-grained striping. Instead, enable an LVM policy that distributes data across large extents on each LUN to ensure each datafile is spread across all available LUNs.

13.7 Datafile Block Size

Some OSs offer a choice of file system block sizes. For file systems supporting datafiles, the block size should be 8KB when compression is used. When compression is not required, a block size of either 8KB or 4KB can be used.

Some OSs offer a choice of file system block sizes. For file systems supporting datafiles, the block size should be 4KB. If a datafile is placed on a file system with a 512-byte block, misaligned files are possible. The LUN and the file system might be properly aligned based on NetApp recommendations, but the file I/O would be misaligned. Such a misalignment would cause severe performance problems.

See additional information on the relationship between block sizes and compression in the section "ONTAP 8.3.10."

13.8 Redo Block Size

File systems supporting redo logs must use a block size that is a multiple of the redo block size. This generally requires that both the redo log file system and the redo log itself use a block size of 512 bytes. At very high redo rates, it is possible that 4KB block sizes perform better, because high redo rates allow I/O to be performed in fewer and more efficient operations. If redo rates are greater than 50MBps, consider testing a 4KB block size.

A few customer problems have been identified with databases using redo logs with a 512-byte block size on a file system with a 4KB block size and many very small transactions. The overhead involved in applying multiple 512-byte changes to a single 4KB file system block led to performance problems that were resolved by changing the file system to use a block size of 512 bytes.

NetApp recommends the following:

- Do not change the redo block size unless advised by a relevant customer support or professional services organization or the change is based on official product documentation.

14 Virtualization

14.1 Overview

Virtualization of databases with VMware ESX, Oracle OVM, or KVM is an increasingly common choice for NetApp customers who chose virtualization for even their most mission-critical databases.

Many misconceptions exist on the support policies for virtualization, particularly for VMware products. Indeed, it is not uncommon to hear that Oracle does not support virtualization in any way. This notion is incorrect and leads to missed opportunities for virtualization. Oracle Doc ID 249212.1 discusses known issues in an Oracle environment and also specifies support for RAC.

A customer with a problem unknown to Oracle might be asked to reproduce the problem on physical hardware. An Oracle customer running a bleeding-edge version of a product might not want to use virtualization because of the potential for new bug discovery. However, this situation has not been a problem in practice for virtualization customers using generally available product versions.

14.2 Storage Presentation

Customers considering virtualization of their databases should base their storage decisions on their business needs. Although this is a generally true statement for all IT decisions, it is especially important for virtualization, because the size and scope of projects vary considerably.

Regarding storage presentation, a storage resource should be managed directly by the VM guest. Therefore, use one of the following storage configurations:

- iSCSI LUNs managed by the iSCSI initiator on the VM, not the hypervisor
- NFS file systems mounted by the VM, not a virtual machine disk (VMDK)
- FC raw device mappings (RDMs) when the VM guest manages the file system

As a general rule, avoid using datastores for Oracle files. There are many reasons for this recommendation:

- **Transparency.** When a VM owns its file systems, it is easier for a database administrator or a system administrator to identify the source of the file systems for their data.
- **Performance.** Testing has shown that there is a performance effect from channeling all I/O through a hypervisor datastore.
- **Manageability.** When a VM owns its file systems, the use or nonuse of a hypervisor layer affects manageability. The same procedures for provisioning, monitoring, data protection, and so on can be used across the entire estate, including both virtualized and nonvirtualized environments.
- **Stability and troubleshooting.** When a VM owns its file systems, delivering good, stable performance and troubleshooting problems are much simpler because the entire storage stack is present on the VM. The hypervisor's only role is to transport FC or IP frames. When a datastore is included in a configuration, it complicates the configuration by introducing another set of timeouts, parameters, log files, and potential bugs.
- **Portability.** When a VM owns its file systems, the process of moving an Oracle environment becomes much simpler. File systems can easily be moved between virtualized and nonvirtualized guests.
- **Vendor lock-in.** After data is placed in a datastore, leveraging a different hypervisor or take the data out of the virtualized environment entirely becomes very difficult.
- **Snapshot enablement.** In some cases, backups in a virtualized environment can become a problem because of the relatively limited bandwidth. For example, a four-port 10GbE trunk might be sufficient to support day-to-day performance needs of many virtualized databases. However, such a trunk would be insufficient to perform backups using RMAN or other backup products that require streaming a full-sized copy of the data.

  Using VM-owned file systems makes it easier to leverage Snapshot-based backups and restores. A VM-owned file system offloads the work of performing backups onto the storage system. There is no need to overbuild the hypervisor configuration purely to support the bandwidth and CPU requirements in the backup window.

NetApp recommends the following:

- For optimum performance and manageability, avoid placing Oracle data on a datastore. Use guest-owned file systems such as NFS or iSCSI file systems managed by the guest or with RDMs.

14.3 Paravirtualized Drivers

For optimum performance, the use of par virtualized network drivers is critical. When a datastore is used, a paravirtualized SCSI driver is required. A paravirtualized device driver allows a guest to integrate more deeply into the hypervisor, as opposed to an emulated driver in which the hypervisor spends more CPU time mimicking the behavior of physical hardware.

The performance of most databases is limited by storage. Therefore, the extra latency introduced by a network or SCSI driver is particularly noticeable. NetApp Customer Support has encountered many performance complaints that were resolved by installing paravirtualized drivers. During one customer proof of concept, databases showed better performance under ESX than with the same hardware running as bare metal. The tests were very I/O intensive, and the performance difference was attributed to the use of the ESX paravirtualized network drivers.

NetApp recommends the following:

- Always use paravirtualized network drivers and SCSI drivers.

14.4 Overcommitting RAM

Overcommitting RAM means configuring more virtualized RAM on various hosts than exists on the physical hardware. Doing so can cause unexpected performance problems. When virtualizing a database, the underlying blocks of the Oracle SGA must not be swapped out to disk by the hypervisor. Doing so causes highly unstable performance results.

NetApp recommends the following**:**

- Do not configure a hypervisor in a way that allows Oracle SGA blocks to be swapped out.

15 Clustering

15.1 Oracle Real Application Clusters

This section applies to Oracle 10.2.0.2 and later. For earlier versions of Oracle, consult Oracle Doc ID 294430.1 in conjunction with this document to determine optimal settings.

disktimeout

The primary storage-related RAC parameter is `disktimeout`. This parameter controls the threshold within which voting file I/O must complete. If the `disktimeout` parameter is exceeded, the RAC node is evicted from the cluster. The default for this parameter is 200. This value should be sufficient for standard storage takeover and giveback procedures.

NetApp strongly recommends testing RAC configurations thoroughly before placing them into production because many factors affect a takeover or giveback. In addition to the time required for storage failover to complete, additional time is also required for Link Aggregation Control Protocol (LACP) changes to propagate, SAN multipathing software must detect an I/O timeout and retry on an alternate path, and, if a database is extremely active, a large amount of I/O must be queued and retried before voting disk I/O is processed.

If an actual storage takeover or giveback cannot be performed, the effect can be simulated with cable pull tests on the database server.

NetApp recommends the following:

- Leave the `disktimeout` parameter at the default value of 200.
- Always test a RAC configuration thoroughly.

misscount

The `misscount` parameter normally affects only the network heartbeat between RAC nodes. The default is 30 seconds. If the grid binaries are on a storage array or the OS boot disk is not local, this parameter might become important. This includes hosts with boot disks located on an FC SAN, NFS-booted OSs, and boot disks located on virtualization datastores such as a VMDK file.

If access to a boot disk is interrupted by a storage takeover or giveback, it is possible that the grid binary location or the entire OS temporarily hangs. The time required for ONTAP to complete the storage operation and for the OS to change paths and resume I/O might exceed the `misscount` threshold. As a result, a node immediately evicts after connectivity to the boot LUN or grid binaries is restored. In most cases, the eviction and subsequent reboot occur with no logging messages to indicate the reason for the reboot. Not all configurations are affected, so test any SAN-booting, NFS-booting, or datastore-based host in a RAC environment so that RAC remains stable if communication to the boot disk is interrupted.

In the case of nonlocal boot disks or the file system hosting `grid` binaries, `misscount` might need to be changed to match `disktimeout`. If this parameter is changed, conduct further testing to also identify any effects on RAC behavior, such as node failover time.

NetApp recommends the following:

- Leave the `misscount` parameter at the default value of 30 unless one of the following conditions applies:
  - `grid` binaries are located on a network-attached disk, including NFS, iSCSI, FC, and datastore-based disks
  - The OS is SAN booted
- In such cases, evaluate the effect of network interruptions that affect access to OS or `GRID_HOME` file systems. In some cases, such interruptions cause the Oracle RAC daemons to stall, which can lead to a `misscount`-based timeout and eviction. The timeout defaults to 27 seconds, which is the value of `misscount` minus `reboottime`. In such cases, increase `misscount` to 200 to match `disktimeout`.

15.2 Solaris Clusters

Solaris clusters, an active-passive clustering technology, are much more highly integrated than other clusterware options. This technology provides an almost plug-and-play capability to easily deployed databases and applications as clustered resources and allows them to be easily moved around the cluster (including associated IP addresses, configuration files, and storage resources). As a result of this tight integration, Oracle has a rigid qualification procedure for Solaris clusters to make sure that all of the components work properly together.

ONTAP provides broad support for Solaris clusters in a SAN environment. Consult the Interoperability Matrix Tool (IMT) for further information.

In an NFS environment, support is limited. There is no supportability barrier with NFS in general, (for example, the use of automounted NFS home directories), but databases cannot be placed under control of Solaris clusters. Previously, an NFS agent was available, but support for this product ended in October of 2012. Although it is possible to use the native ability of Solaris clusters to build a custom service that can be clustered, this is probably not feasible for most deployments. The reason is the time and effort required to write scripts that manage resources, including storage.

15.3 Veritas Cluster Server

Veritas Cluster Server (VCS) is similar to Solaris clusters in that it allows users to package a database or application as a deployed service and deploy it on a cluster in an active-passive manner.

VCS and SAN

ONTAP provides broad support for VCS clustering in a SAN environment. Consult the Interoperability Matrix Tool (IMT) for further information.

VCS and NFS

At one time, an NFS client was available from NetApp to provide quorum, monitoring, management, fencing, and NFS lock-breaking capabilities. However, support was discontinued in October of 2012 primarily because these capabilities were no longer required. VCS can now natively manage and monitor NFS file systems. Multiple options exist for quorum management in a NAS environment that do not require an agent.

VCS and NFS Fencing

One consideration for any active-passive clustering is fencing, which means that storage resources are available to only one node in the cluster. In a SAN environment, fencing usually means using SCSI persistent reservations, which allows a node to claim exclusive control of a LUN. In an NFS context, it means changing the export options for a file system to make it impossible to access a resource on more than one node. The difference is that, in a SAN environment, fencing is performed by a node in the cluster laying claim to a storage resource. In a NAS environment, the fencing must be performed on the storage system.

Fencing with NFS is not strictly necessary. It is much more important to have fencing in a SAN environment because the simple act of mounting a SAN file system on more than one server generally corrupts data immediately. NFS is a clustered file system, which means that multiple servers can mount a file system without problems.

Many customers use active-passive clustering with VCS and similar products such as HP ServiceGuard and IBM PowerHA without any fencing in place. They trust the cluster software itself to make sure that a resource runs on only one node. When fencing is desired, it can be deployed as part of the cluster resource with a small scripting effort.

When a service starts up, it issues a command to the storage system to (a) cut off access for the target file systems to all nodes and then (b) grant access to the one node on which the service starts. Therefore, only one node is able to perform I/O on the target file systems. When a service shuts down, it issues a command to the storage system to remove its access. Other variants exist, but this is the most comprehensive approach.

Assistance with these systems is available from NetApp Professional Services. Contact your NetApp representative for more information.

VCS and NFS Lock Breaking

NFS locks in an Oracle environment are a form of fencing. An Oracle database does not start if it finds an NFS lock in place on the target files. In a VCS environment, NFS locking generally interferes with the normal functioning of the VCS cluster. The only time locks must be broken is when one node takes over the services of another node that has not shut down gracefully. During a clean shutdown of an Oracle database, locks are removed. If the node crashes, locks are left in place and must be cleared before the database can be restarted.

Most customers choose to disable NFS locking by including the appropriate NFS mount option that prevents locks from being created in the first place. If this is not desirable, lock breaking can be scripted. As with fencing, assistance for lock-break scripting is available from NetApp Professional Services, and, in some cases, fully supported options might be available through Rapid Response Engineering. Contact your NetApp representative for more information.

16 IBM AIX

This section addresses configuration topics specific to the IBM AIX operating system.

16.1 Concurrent I/O

Achieving optimum performance on IBM AIX requires the use of concurrent I/O. Without concurrent I/O, performance limitations are likely because AIX performs serialized, atomic I/O, which incurs significant overhead.

Originally, NetApp recommended using the `cio` mount option to force the use of concurrent I/O on the file system, but this process had drawbacks and is no longer required. Since the introduction of AIX 5.2

and Oracle 10gR1, Oracle on AIX can open individual files for concurrent IO, as opposed to forcing concurrent I/O on the entire file system.

The best method for enabling concurrent I/O is to set the `init.ora` parameter `filesystemio_options` to `setall`. Doing so allows Oracle to open specific files for use with concurrent I/O.

Using `cio` as a mount option forces the use of concurrent I/O, which can have negative consequences. For example, forcing concurrent I/O disables readahead on file systems, which can damage performance for I/O occurring outside the Oracle database software, such as copying files and performing tape backups. Furthermore, products such as Oracle GoldenGate and SAP BR*Tools are not compatible with using the `cio` mount option with certain versions of Oracle.

NetApp recommends the following:

- Do not use the `cio` mount option at the file system level. Rather, enable concurrent I/O through the use of `filesystemio_options=setall`.
- Only use the `cio` mount option should if it is not possible to set `filesystemio_options=setall`.

16.2 AIX NFSv3 Mount Options

Table 1 and Table 2 list the AIX NFSv3 mount options.

Table 1) AIX NFSv3 mount options—single instance.

File Type	Mount Options
`ADR_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
Control files Datafiles Redo logs	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
`ORACLE_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,intr

Table 2) AIX NFSv3 mount options—RAC.

File Type	Mount Options
`ADR_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
Control files Datafiles Redo logs	rw,bg,hard,vers=3,proto=cp,timeo=600,rsize=65536,wsize=65536,nointr, noac
CRS/Voting	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac
Dedicated `ORACLE_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
Shared `ORACLE_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr

The primary difference between single-instance and RAC mount options is the addition of `noac` to the mount options. This addition has the effect of disabling the host OS caching that enables all instances in the RAC cluster to have a consistent view of the state of the data.

Although using the `cio` mount option and the `init.ora` parameter `filesystemio_options=setall` has the same effect of disabling host caching, it is still necessary to use `noac`. `Noac` is required for shared `ORACLE_HOME` deployments to facilitate the consistency of files such as Oracle password files and

spfile parameter files. If each instance in a RAC cluster has a dedicated ORACLE_HOME, then this parameter is not required.

16.3 AIX jfs/jfs2 Mount Options

Table 3 lists the AIX jfs/jfs2 mount options.

Table 3) AIX jfs/jfs2 mount options—single instance.

File Type	Mount Options
ADR_HOME	Defaults
Control files Data files Redo logs	Defaults
ORACLE_HOME	Defaults

Before using AIX hdisk devices in any environment, including databases, check the parameter queue_depth. This parameter is not the HBA queue depth; rather it relates to the SCSI queue depth of the individual hdisk device. Depending on how the LUNs are configured, the value for queue_depth might be too low for good performance. Testing has shown the optimum value to be 64.

17 HP-UX

This section addresses configuration topics specific to the HP-UX operating system.

17.1 HP-UX NFSv3 Mount Options

Table 4 lists the HP-UX NFSv3 mount options.

Table 4) HP-UX NFSv3 mount options—single instance.

File Type	Mount Options
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid
Control files Datafiles Redo logs	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,forcedirectio,nointr,suid
ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid

Table 5) HP-UX NFSv3 mount options—RAC

File Type	Mount Options
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,noac,suid
Control files Datafiles Redo logs	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac forcedirectio, suid
CRS/Voting	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac, forcedirectio,suid

File Type	Mount Options
Dedicated ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid
Shared ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac,suid

The primary difference between single-instance and RAC mount options is the addition of noac and forcedirectio to the mount options. This addition has the effect of disabling host OS caching, which enables all instances in the RAC cluster to have a consistent view of the state of the data. Although using the init.ora parameter filesystemio_options=setall has the same effect of disabling host caching, it is still necessary to use noac and forcedirectio.

The reason noac is required for shared ORACLE_HOME deployments is to facilitate consistency of files such as Oracle password files and spfiles. If each instance in a RAC cluster has a dedicated ORACLE_HOME, this parameter is not required.

17.2 HP-UX VxFS Mount Options

Use the following mount options for file systems hosting Oracle binaries:

```
delaylog,nodatainlog
```

Use the following mount options for file systems containing datafiles, redo logs, archive logs, and control files in which the version of HP-UX does not support concurrent I/O:

```
nodatainlog,mincache=direct,convosync=direct
```

When concurrent I/O is supported (VxFS 5.0.1 and later, or with the ServiceGuard Storage Management Suite), use these mount options for file systems containing datafiles, redo logs, archive logs, and control files:

```
delaylog,cio
```

Note: The parameter db_file_multiblock_read_count is especially critical in VxFS environments. Oracle recommends that this parameter remain unset in Oracle 10g R1 and later unless specifically directed otherwise. The default with an Oracle 8KB block size is 128. If the value of this parameter is forced to 16 or less, remove the convosync=direct mount option because it can damage sequential I/O performance. This step damages other aspects of performance and should only be taken if the value of db_file_multiblock_read_count must be changed from the default value.

18 Linux

This section addresses configuration topics specific to the Linux OS.

18.1 Linux NFS

Slot Tables

NFS performance on Linux depends on a parameter called tcp_slot_table_entries. This parameter regulates the number of outstanding NFS operations that are permitted on a Linux OS.

The default in most 2.6-derived kernels, which includes RH5 and OL5, is 16, and this default frequently causes performance problems. The opposite problem occurs on newer kernels in which the tcp_slot_table_entries value is uncapped and can cause storage problems by flooding the system with excessive requests.

The solution is to set this value statically. Use a value of 128 for any Linux OS using NetApp NFS storage with an Oracle database.

To set this value in RHEL 6.2 and earlier, place the following entry in `/etc/sysctl.conf`:

```
sunrpc.tcp_slot_table_entries = 128
```

In addition, there is a bug in most Linux distributions using 2.6 kernels. The startup process reads the contents of `/etc/sysctl.conf` before the NFS client is loaded. As a result, when the NFS client is eventually loaded, it takes the default value of 16. To avoid this problem, edit `/etc/init.d/netfs` to call `/sbin/sysctl -p` in the first line of the script so that `tcp_slot_table_entries` is set to 128 before NFS mounts any file systems.

To set this value in RHEL 6.3 and later, apply the following modification in the RPC configuration file:

```
echo "options sunrpc udp_slot_table_entries=64 tcp_slot_table_entries=128
tcp_max_slot_table_entries=128" >> /etc/modprobe.d/sunrpc.conf
```

18.2 Linux NFSv3 Mount Options

Table 6 and Table 7 list the Linux NFSv3 mount options.

Table 6) Linux NFSv3 mount options—single instance.

File Type	Mount Options
`ADR_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
Control files Datafiles Redo logs	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr
`ORACLE_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr

Table 7) Linux NFSv3 mount options—RAC.

File Type	Mount Options
`ADR_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,actimeo=0
Control files Data files Redo logs	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,actimeo=0
CRS/voting	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac,actimeo=0
Dedicated `ORACLE_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
Shared `ORACLE_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,actimeo=0

The primary difference between single-instance and RAC mount options is the addition of `actimeo=0` to the mount options. This addition has the effect of disabling the host OS caching, which enables all instances in the RAC cluster to have a consistent view of the state of the data. Although using the `init.ora` parameter `filesystemio_options=setall` has the same effect of disabling host caching, it is still necessary to use `actimeo=0`.

The reason `actimeo=0` is required for shared `ORACLE_HOME` deployments is to facilitate consistency of files such as the Oracle password files and spfiles. If each instance in a RAC cluster has a dedicated `ORACLE_HOME`, then this parameter is not required.

Generally, nondatabase files should be mounted with the same options used for single-instance datafiles, although specific applications might have different requirements. Avoid the mount options `noac` and `actimeo=0` if possible because these options disable file system-level readahead and buffering. This can cause severe performance problems for processes such as extraction, translation, and loading.

ACCESS and GETATTR

Some customers have noted that an extremely high level of other IOPS such as ACCESS and GETATTR can dominate their workloads. In extreme cases, operations such as reads and writes can be as low as 10% of the total. This is normal behavior with any database that includes using `actimeo=0` and/or `noac` on Linux because these options cause the Linux OS to constantly reload file metadata from the storage system. Operations such as ACCESS and GETATTR are low-impact operations that are serviced from the ONTAP cache in a database environment. They should not be considered genuine IOPS, such as reads and writes, that create true demand on storage systems. These other IOPS do create some load, however, especially in RAC environments. To address this situation, enable DNFS, which bypasses the OS buffer cache and avoids these unnecessary metadata operations.

Linux Direct NFS

One additional mount option, called `nosharecache`, is required when (a) DNFS is enabled and (b) a source volume is mounted more than once on a single server (c) with a nested NFS mount. This configuration is seen primarily in environments supporting SAP applications. For example, a single volume on a NetApp system could have a directory located at `/vol/oracle/base` and a second at `/vol/oracle/home`. If `/vol/oracle/base` is mounted at `/oracle` and `/vol/oracle/home` is mounted at `/oracle/home`, the result is nested NFS mounts that originate on the same source.

The OS can detect the fact that `/oracle` and `/oracle/home` reside on the same volume, which is the same source file system. The OS then uses the same device handle for accessing the data. Doing so improves the use of OS caching and certain other operations, but it interferes with DNFS. If DNFS must access a file, such as the `spfile`, on `/oracle/home`, it might erroneously attempt to use the wrong path to the data. The result is a failed I/O operation. In these configurations, add the `nosharecache` mount option to any NFS file system that shares a source FlexVol volume with another NFS file system on that host. Doing so forces the Linux OS to allocate an independent device handle for that file system.

Linux Direct NFS and Oracle RAC

The use of DNFS has special performance benefits for Oracle RAC on the Linux OS because Linux does not have a method to force direct I/O, which is required with RAC for coherency across the nodes. As a workaround, Linux requires the use of the `actimeo=0` mount option, which causes file data to expire immediately from the OS cache. This option in turn forces the Linux NFS client to constantly reread attribute data, which damages latency and increases load on the storage controller.

Enabling DNFS bypasses the host NFS client and avoids this damage. Multiple customers have reported significant performance improvements on RAC clusters and significant decreases in ONTAP load (especially with respect to other IOPS) when enabling DNFS.

Linux Direct NFS and oranfstab File

When using DNFS on Linux with the multipathing option, multiple subnets must be used. On other OSs, multiple DNFS channels can be established by using the `LOCAL` and `DONTROUTE` options to configure multiple DNFS channels on a single subnet. However, this does not work properly on Linux and

unexpected performance problems can result. With Linux, each NIC used for DNFS traffic must be on a different subnet.

18.3 General Linux SAN Configuration

Compression Alignment—Partitions

Compression requires alignment to 8KB disk boundaries for optimum results. Check alignment by using the fdisk utility with the `-u` option to view a disk based in sectors. See the following example:

```
[root@jfs0 etc]# fdisk -l -u /dev/sdb

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
64 heads, 32 sectors/track, 10240 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 65536 bytes
Disk identifier: 0xb97f94c1

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1              36    20971519    10485742   83  Linux
Partition 1 does not start on physical sector boundary.
```

This partition is not 8KB aligned. Rather, the partition has an offset of 36 sectors. This offset aligns to a 4KB boundary, which is generally required for good performance but does not align to an 8KB boundary. The start of a partition should be a multiple of 16 sectors (512 bytes * 16 = 8192) so that the partition is aligned.

This example shows a correctly aligned partition:

```
[root@jfs0 etc]# fdisk -l -u /dev/sdb

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
64 heads, 32 sectors/track, 10240 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 65536 bytes
Disk identifier: 0xb97f94c1

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1              64    20971519    10485728   83  Linux
```

Compression Alignment—File Systems

In addition to the partition, the file system must also be aligned to 8KB boundaries. This means that the block size of the file system must be 8KB. When using Oracle ASM, 8KB alignment is ensured because of the way Oracle ASM performs extent allocation and striping.

When using other file systems, the block size must be specified at 8KB. Doing so might not be possible with all file systems.

I/O Scheduler

The Linux kernel allows low-level control over the way that I/O to block devices is scheduled. The defaults on various distribution of Linux vary considerably. Testing shows that Deadline usually offers the best results, but on occasion NOOP has been slightly better. The difference in performance is minimal, but test both options if it is necessary to extract the maximum possible performance from a database configuration. CFQ is the default in many configurations, and it has demonstrated significant performance problems with database workloads.

See the relevant Linux vendor documentation for instructions on configuring the I/O scheduler.

Multipathing

Some customers have encountered crashes during network disruption because the multipath daemon was not running on their system. On recent versions of Linux, the installation process of the OS and the multipathing daemon might leave these OSs vulnerable to this problem. The packages are installed correctly, but they are not configured for automatic startup after a reboot.

For example, the default for the multipath daemon on RHEL5.5 might appear as follows:

```
[root@jfs0 iscsi]# chkconfig --list | grep multipath
multipathd      0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

This can be corrected with the following commands:

```
[root@jfs0 iscsi]# chkconfig multipathd on
[root@jfs0 iscsi]# chkconfig --list | grep multipath
multipathd      0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

18.4 ASM Mirroring

ASM mirroring might require changes to the Linux multipath settings to allow ASM to recognize a problem and switch over to an alternate fail group. Most ASM configurations on ONTAP use external redundancy, which means that data protection is provided by the external array and ASM does not mirror data. Some sites use ASM with normal redundancy to provide two-way mirroring, normally across different sites.

The Linux settings shown in the NetApp Host Utilities documentation include multipath parameters that result in indefinite queuing of I/O. This means an I/O on a LUN device with no active paths waits as long as required for the I/O to complete. This is usually desirable because Linux hosts wait as long as needed for SAN path changes to complete, for FC switches to reboot, or for a storage system to complete a failover.

This unlimited queuing behavior causes a problem with ASM mirroring because ASM must receive an I/O failure for it to retry I/O on an alternate LUN.

Set the following parameters in the Linux `multipath.conf` file for ASM LUNs used with ASM mirroring:

```
polling_interval 5
no_path_retry  24
```

These settings create a 120-second timeout for ASM devices. The timeout is calculated as the `polling_interval` * `no_path_retry` as seconds. The exact value might need to be adjusted in some circumstances, but a 120 second timeout should be sufficient for most uses. Specifically, 120 seconds should allow a controller takeover or giveback to occur without producing an I/O error that would result in the fail group being taken offline.

A lower `no_path_retry` value can shorten the time required for ASM to switch to an alternate fail group, but this also increases the risk of an unwanted failover during maintenance activities such as a controller takeover. The risk can be mitigated by careful monitoring of the ASM mirroring state. If an unwanted failover occurs, the mirrors can be rapidly resynced if the resync is performed relatively quickly. For additional information, see the Oracle documentation on ASM Fast Mirror Resync for the version of Oracle software in use.

18.5 ASMlib Block Sizes

ASMlib is an optional ASM management library and associated utilities. Its primary value is the capability to stamp a LUN or an NFS-based file as an ASM resource with a human-readable label.

Recent versions of ASMlib detect a LUN parameter called Logical Blocks Per Physical Block Exponent (LBPPBE). This value was not reported by the ONTAP SCSI target until recently. It now returns a value that indicates that a 4KB block size is preferred. This is not a definition of block size, but it is a hint to any

application that uses LBPPBE that I/Os of a certain size might be handled more efficiently. ASMlib does, however, interpret LBPPBE as a block size and persistently stamps the ASM header when the ASM device is created.

This process can cause problems with upgrades and migrations in a number of ways, all based on the inability to mix ASMlib devices with different block sizes in the same ASM disk group.

For example, older arrays generally reported an LBPPBE value of 0 or did not report this value at all. ASMlib interprets this as a 512-byte block size. Newer arrays would be interpreted as having a 4KB block size. It is not possible to mix both 512-byte and 4KB devices in the same ASM disk group. Doing so would block a user from increasing the size of the ASM disk group using LUNs from two arrays or leveraging ASM as a migration tool. In other cases, RMAN might not permit the copying of files between an ASM disk group with a 512-byte block size and an ASM disk group with a 4KB block size.

The preferred solution is to patch ASMlib. The Oracle bug ID is 13999609, and the patch is present in oracleasm-support-2.1.8-1 and higher. This patch allows a user to set the parameter `ORACLEASM_USE_LOGICAL_BLOCK_SIZE` to `FALSE` in the `/etc/sysconfig/oracleasm` configuration file. Doing so blocks ASMlib from using the LBPPBE parameter, which means that LUNs on the new array are now recognized as 512-byte block devices.

Note: The option does not change the block size on LUNs that were previously stamped by ASMlib. For example, if an ASM disk group with 512-byte blocks must be migrated to a new storage system that reports a 4KB block, the option `ORACLEASM_USE_LOGICAL_BLOCK_SIZE` must be set before the new LUNs are stamped with ASMlib.

If ASMlib cannot be patched, ASMlib can be removed from the configuration. This change is disruptive and requires the unstamping of ASM disks and making sure that the `asm_diskstring` parameter is set correctly. This change does not, however, require the migration of data.

18.6 Linux ext3 and ext4 Mount Options

NetApp recommends using the default mount options.

19 Microsoft Windows

This section addresses configuration topics specific to the Microsoft Windows OS.

19.1 NFS

Oracle supports the use of Microsoft Windows with the direct NFS client. This capability offers a path to the management benefits of NFS, including the ability to view files across environments, dynamically resize volumes, and leverage a less expensive IP protocol. See the official Oracle documentation for information on installing and configuring a database on Microsoft Windows using DNFS. No special best practices exist.

19.2 SAN

For optimum compression efficiency, make sure that NTFS file systems use an 8192 byte or larger allocation unit. Use of a 4096-byte allocation unit, which is generally the default, damages efficiency.

20 Solaris

This section addresses configuration topics specific to the Solaris OS.

20.1 Solaris NFSv3 Mount Options

Table 8 lists the Solaris NFSv3 mount options for a single instance.

Table 8) Solaris NFSv3 mount options—single instance.

File Type	Mount Options
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
Control files Datafiles Redo logs	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,llock,suid
ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid

The use of `llock` has been proven to dramatically improve performance in customer environments by removing the latency associated with acquiring and releasing locks on the storage system. Use this option with care in environments in which numerous servers are configured to mount the same file systems and Oracle is configured to mount these databases. Although this is a highly unusual configuration, it is used by a small number of customers. If an instance is accidentally started a second time, data corruption can occur because Oracle is unable to detect the lock files on the foreign server. NFS locks do not otherwise offer protection; as in NFS version 3, they are advisory only.

Because the `llock` and `forcedirectio` parameters are mutually exclusive, it is important that `filesystemio_options=setall` is present in the `init.ora` file so that `directio` is used. Without this parameter, host OS buffer caching is used and performance can be adversely affected.

Table 9 lists the Solaris NFSv3 RAC mount options.

Table 9) Solaris NFSv3 mount options—RAC.

File Type	Mount Options
ADR_HOME	rw, bg, hard, vers=3, proto=tcp, timeo=600, rsize=65536, wsize=65536, noac
Control files Data files Redo logs	rw, bg, hard, vers=3, proto=tcp, timeo=600, rsize=65536, wsize=65536, nointr, noac, forcedirectio
CRS/Voting	rw, bg, hard, vers=3, proto=tcp, timeo=600, rsize=65536, wsize=65536, nointr, noac, forcedirectio
Dedicated ORACLE_HOME	rw, bg, hard, vers=3, proto=tcp, timeo=600, rsize=65536, wsize=65536, suid
Shared ORACLE_HOME	rw, bg, hard, vers=3, proto=tcp, timeo=600, rsize=65536, wsize=65536, nointr, noac, suid

The primary difference between single-instance and RAC mount options is the addition of `noac` and `forcedirectio` to the mount options. This addition has the effect of disabling the host OS caching, which enables all instances in the RAC cluster to have a consistent view of the state of the data. Although using the `init.ora` parameter `filesystemio_options=setall` has the same effect of disabling host caching, it is still necessary to use `noac` and `forcedirectio`.

The reason `actimeo=0` is required for shared `ORACLE_HOME` deployments is to facilitate consistency of files such as Oracle password files and spfiles. If each instance in a RAC cluster has a dedicated `ORACLE_HOME`, this parameter is not required.

20.2 Solaris UFS Mount Options

NetApp strongly recommends using the logging mount option so that data integrity is preserved in the case of a Solaris host crash or the interruption of FC connectivity. The logging mount option also preserves the usability of Snapshot backups.

20.3 Solaris ZFS

Solaris ZFS must be installed and configured carefully to deliver optimum performance.

mvector

Solaris 11 included a change in how it processes large IO operations which can result in severe performance problems on SAN storage arrays. The problem is documented in detail in the NetApp bug report 630173, "Solaris 11 ZFS Performance Regression." The solution is to change an OS parameter called `zfs_mvector_max_size`.

Run the following command as root:

```
echo "zfs_mvector_max_size/W 0t131072" |mdb -kw
```

If any unexpected problems arise from this change, it can be easily reversed by running the following command as root:

```
echo "zfs_mvector_max_size/W 0t1048576" |mdb -kw
```

Kernel

Reliable ZFS performance requires a Solaris kernel patched against LUN alignment problems. The fix was introduced with patch 147440-19 in Solaris 10 and with SRU 10.5 for Solaris 11. Only use Solaris 10 and later with ZFS.

LUN Configuration

To configure a LUN, complete the following steps:

1. Create a LUN of type `solaris`.
2. Install the appropriate Host Utility Kit (HUK) specified by the IMT.
3. Follow the instructions in the HUK exactly as described. The basic steps are outlined in this section, but refer to the latest documentation for the proper procedure.
    a. Run the `host_config` utility to update the `sd.conf/sdd.conf` file. Doing so allows the SCSI drives to correctly discover ONTAP LUNs.
    b. Follow the instructions given by the `host_config` utility to enable multipath input/output (MPIO).
    c. Reboot. This step is required so that any changes are recognized across the system.
4. Partition the LUNs and verify that they are properly aligned. See "Appendix B: WAFL Alignment Verification" for instructions on how to directly test and confirm alignment.

zpools

A zpool should only be created after the steps in the section “LUN Configuration” are performed. If the procedure is not done correctly, it can result in serious performance degradation due to the I/O alignment. Optimum performance on ONTAP requires I/O to be aligned to a 4K boundary on disk. The file systems created on a zpool will use an effective block size that is controlled through a parameter called `ashift`, which can be viewed by running the command `zdb -C`.

The value of `ashift` defaults to 9, which means 2^9, or 512 bytes. For optimum performance, the `ashift` value must be 12 (2^12=4K). This value is set at the time the zpool is created and cannot be changed, which means that data in zpools with `ashift` other than 12 should be migrated by copying data to a newly created zpool.

After creating a zpool, verify the value of `ashift` before proceeding. If the value is not 12, the LUNs were not discovered correctly. Destroy the zpool, verify that all steps shown in the relevant Host Utilities documentation were performed correctly, and recreate the zpool.

zpools and Solaris LDOMs

Solaris LDOMs create an additional requirement for making sure that I/O alignment is correct. Although a LUN might be properly discovered as a 4K device, a virtual vdsk device on an LDOM does not inherit the configuration from the I/O domain. The vdsk based on that LUN will default back to a 512-byte block.

An additional configuration file is required. First, the individual LDOM's must be patched for Oracle bug 15824910 to enable the additional configuration options. This patch has been ported into all currently used versions of Solaris. Once the LDOM is patched, it is ready for configuration of the new properly aligned LUNs as follows:

1. Identify the LUN or LUNs to be used in the new zpool. In this example, it is the c2d1 device.

```
root@LDOM1 # echo | format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
  0. c2d0 <Unknown-Unknown-0001-100.00GB>
     /virtual-devices@100/channel-devices@200/disk@0
  1. c2d1 <SUN-ZFS Storage 7330-1.0 cyl 1623 alt 2 hd 254 sec 254>
     /virtual-devices@100/channel-devices@200/disk@1
```

2. Retrieve the vdc instance of the devices to be used for a ZFS pool:

```
root@LDOM1 # cat /etc/path_to_inst
#
# Caution! This file contains critical kernel state
#
"/fcoe" 0 "fcoe"
"/iscsi" 0 "iscsi"
"/pseudo" 0 "pseudo"
"/scsi_vhci" 0 "scsi_vhci"
"/options" 0 "options"
"/virtual-devices@100" 0 "vnex"
"/virtual-devices@100/channel-devices@200" 0 "cnex"
"/virtual-devices@100/channel-devices@200/disk@0" 0 "vdc"
"/virtual-devices@100/channel-devices@200/pciv-communication@0" 0 "vpci"
"/virtual-devices@100/channel-devices@200/network@0" 0 "vnet"
"/virtual-devices@100/channel-devices@200/network@1" 1 "vnet"
"/virtual-devices@100/channel-devices@200/network@2" 2 "vnet"
"/virtual-devices@100/channel-devices@200/network@3" 3 "vnet"
"/virtual-devices@100/channel-devices@200/disk@1" 1 "vdc" << We want this one
```

3. Edit the `/platform/sun4v/kernel/drv/vdc.conf`:

```
block-size-list="1:4096";
```

   This means that device instance 1 will be assigned a block size of 4096.

   As an additional example, assume vdsk instances 1 through 6 need to be configured for a 4K block size and `/etc/path_to_inst` reads as follows:

```
"/virtual-devices@100/channel-devices@200/disk@1" 1 "vdc"
"/virtual-devices@100/channel-devices@200/disk@2" 2 "vdc"
"/virtual-devices@100/channel-devices@200/disk@3" 3 "vdc"
"/virtual-devices@100/channel-devices@200/disk@4" 4 "vdc"
"/virtual-devices@100/channel-devices@200/disk@5" 5 "vdc"
```

```
"/virtual-devices@100/channel-devices@200/disk@6" 6 "vdc"
```

4. The final `vdc.conf` file should contain the following:

```
block-size-list="1:8192","2:8192","3:8192","4:8192","5:8192","6:8192";
```

Caution

The LDOM must be rebooted **after** `vdc.conf` is configured and the vdsk is created. This step cannot be avoided. The block size change only takes effect **after** a reboot. Proceed with zpool configuration and ensure that `ashift` is properly set to 12 as described previously.

ZIL

Generally, there is no reason to locate the ZFS Intent Log (ZIL) on a different device. The log can share space with the main pool. The primary use of a separate ZIL is when using physical drives that lack the write caching features in modern storage arrays.

logbias

Set the `logbias` parameter on ZFS file systems hosting Oracle data.

```
zfs set logbias=throughput <filesystem>
```

Using this parameter reduces overall write levels. Under the defaults, written data is committed first to the ZIL and then to the main storage pool. This approach is appropriate for a configuration using a plain disk configuration, which includes a SSD-based ZIL device and spinning media for the main storage pool, because it allows a commit to occur in a single I/O transaction on the lowest latency media available.

When using a modern storage array that includes its own caching capability, this approach is not generally necessary. Under rare circumstances, it might be desirable to commit a write with a single transaction to the log, such as a workload that consists of highly concentrated, latency-sensitive random writes. There are consequences in the form of write amplification because the logged data is eventually written to the main storage pool, resulting in a doubling of the write activity.

Direct I/O

Many applications, including Oracle products, can bypass the host buffer cache by enabling direct I/O. This strategy does not work as expected with ZFS file systems. Although the host buffer cache is bypassed, ZFS itself continues to cache data. This action can result in misleading results when using tools such as fio or sio to perform performance tests because it is difficult to predict whether I/O is reaching the storage system or whether it is being cached locally within the OS. This action also makes it very difficult to use such synthetic tests to compare ZFS performance to other file systems. As a practical matter, there is little to no difference in file system performance under real user workloads.

Multiple zpools

Snapshot-based backups, restores, clones, and archiving of ZFS-based data must be performed at the level of the zpool and typically requires multiple zpools. A zpool is analogous to an LVM disk pool and should be configured using the same rules. For example, a database is probably best laid out with the datafiles residing on `zpool1` and the archive logs, control files, and redo logs residing on `zpool2`. This approach permits a standard hot backup in which the database is placed in hot backup mode, followed by a Snapshot copy of `zpool1`. The database is then removed from hot backup mode, the log archive is forced, and a Snapshot copy of `zpool2` is created. A restore operation requires unmounting the zfs file systems and offlining the zpool in its entirety, following by a SnapRestore restore operation. The zpool can then be brought online again and the database recovered.

filesystemio_options

The Oracle parameter `filesystemio_options` works differently with ZFS. If `setall` or `directio` is used, write operations are synchronous and bypass the OS buffer cache, but reads are buffered by ZFS. This action causes difficulties in performance analysis because I/O is sometimes intercepted and serviced by the ZFS cache, making storage latency and total I/O less than it might appear to be.

21 Conclusion

As stated at the start of this document, there are few true best practices for an Oracle storage configuration because there is so much variability between implementations. A database project could contain one mission-critical database or it could contain 5000 legacy databases or sizes ranges from a handful of gigabytes to hundreds of terabytes. Options such as clusterware and virtualization introduce further variables.

A better term is design considerations or issues that must be considered while planning a storage implementation. The right solution depends on both the technical details of the implementation and the business requirements driving the project. NetApp and partner professional services experts are available for assistance in complex projects. Even if assistance is not required for the duration of the project, NetApp strongly encourages new customers to use professional services for assistance in developing a high-level approach.

Appendix A: Stale NFS Locks

If an Oracle database server crashes, it might have problems with stale NFS locks upon restart. This problem is avoidable by paying careful attention to the configuration of name resolution on the server.

This problem arises because creating a lock and clearing a lock use two slightly different methods of name resolution. Two processes are involved, the Network Lock Manager (NLM) and the NFS client. The NLM uses `uname -n` to determine the host name, while the `rpc.statd` process uses `gethostbyname()`. These host names must match for the OS to properly clear stale locks. For example, the host might be looking for locks owned by `dbserver5`, but the locks were registered by the host as `dbserver5.mydomain.org`. If `gethostbyname()` does not return the same value as `uname -a`, then the lock release process did not succeed.

The following sample script verifies whether name resolution is fully consistent:

```
#! /usr/bin/perl
$uname=`uname -n`;
chomp($uname);
($name, $aliases, $addrtype, $length, @addrs) = gethostbyname $uname;
print "uname -n yields: $uname\n";
print "gethostbyname yields: $name\n";
```

If `gethostbyname` does not match `uname`, stale locks are likely. For example, this result reveals a potential problem:

```
uname -n yields: dbserver5
gethostbyname yields: dbserver5.mydomain.org
```

The solution is usually found by changing the order in which hosts appear in `/etc/hosts`. For example, assume that the hosts file includes this entry:

```
10.156.110.201  dbserver5.mydomain.org dbserver5 loghost
```

To resolve this issue, change the order in which the fully qualified domain name and the short host name appear:

```
10.156.110.201  dbserver5 dbserver5.mydomain.org loghost
```

`gethostbyname()` now returns the short `dbserver5` host name, which matches the output of `uname`. Locks are thus cleared automatically after a server crash.

Appendix B: WAFL Alignment Verification

Correct WAFL alignment is critical for good performance. Although ONTAP manages blocks in 4KB units, this fact does not mean that ONTAP performs all operations in 4KB units. In fact, ONTAP supports block operations of different sizes, but the underlying accounting is managed by WAFL in 4KB units.

The term “alignment” refers to how Oracle I/O corresponds to these 4KB units. Optimum performance requires an Oracle 8KB block to reside on two 4KB WAFL physical blocks on a disk. If a block is offset by 2KB, this block resides on half of one 4KB block, a separate full 4KB block, and then half of a third 4KB block. This arrangement causes performance degradation.

Alignment is not a concern with NAS file systems. Oracle datafiles are aligned to the start of the file based on the size of the Oracle block. Therefore, block sizes of 8KB, 16KB, and 32KB are always aligned. All block operations are offset from the start of the file in units of 4 kilobytes.

LUNs, in contrast, generally contain some kind of disk header or file system metadata at their start that creates an offset. Alignment is rarely a problem in modern OSs because these OSs are designed for physical disks that might use a native 4KB sector, which also requires I/O to be aligned to 4KB boundaries for optimum performance.

There are, however, some exceptions. A database might have been migrated from an older OS that was not optimized for 4KB I/O, or user error during partition creation might have led to an offset that is not in units of 4KB in size.

The following examples are Linux-specific, but the procedure can be adapted for any OS.

Aligned

The following example shows an alignment check on a single LUN with a single partition.

First, create the partition that uses all partitions available on the disk.

```
[root@jfs0 iscsi]# fdisk /dev/sdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel with disk identifier 0xb97f94c1.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

The device presents a logical sector size that is smaller than
the physical sector size. Aligning to a physical sector (or optimal
I/O) size boundary is recommended, or performance may be impacted.

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-10240, default 1):
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-10240, default 10240):
Using default value 10240

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

```
[root@jfs0 iscsi]#
```

The alignment can be checked mathematically with the following command:

```
[root@jfs0 iscsi]# fdisk -u -l /dev/sdb

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
64 heads, 32 sectors/track, 10240 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 65536 bytes
Disk identifier: 0xb97f94c1

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1              32    20971519    10485744   83  Linux
```

The output shows that the units are 512 bytes, and the start of the partition is 32 units. This is a total of 32 x 512 = 16,834 bytes, which is a whole multiple of 4KB WAFL blocks. This partition is correctly aligned.

To verify correct alignment, complete the following steps:

1. Identify the universally unique identifier (UUID) of the LUN.

```
FAS8040SAP::> lun show -v /vol/jfs_luns/lun0
              Vserver Name: jfs
                  LUN UUID: ed95d953-1560-4f74-9006-85b352f58fcd
                    Mapped: mapped
```

2. Enter the node shell on the ONTAP controller.

```
FAS8040SAP::> node run -node FAS8040SAP-02
Type 'exit' or 'Ctrl-D' to return to the CLI
FAS8040SAP-02> set advanced
set not found.  Type '?' for a list of commands
FAS8040SAP-02> priv set advanced
Warning: These advanced commands are potentially dangerous; use
         them only when directed to do so by NetApp
         personnel.
```

3. Start statistical collections on the target UUID identified in the first step.

```
FAS8040SAP-02*> stats start lun:ed95d953-1560-4f74-9006-85b352f58fcd
Stats identifier name is 'Ind0xffffff08b9536188'
FAS8040SAP-02*>
```

4. Perform some I/O. It is important to use the `iflag` argument to make sure that I/O is synchronous and not buffered.

   Note: Be very careful with this command. Reversing the `if` and `of` arguments destroys data.

```
[root@jfs0 iscsi]# dd if=/dev/sdb1 of=/dev/null iflag=dsync count=1000 bs=4096
1000+0 records in
1000+0 records out
4096000 bytes (4.1 MB) copied, 0.0186706 s, 219 MB/s
```

5. Stop the stats and view the alignment histogram. All I/O should be in the `.0` bucket, which indicates I/O that is aligned to a 4KB block boundary.

```
FAS8040SAP-02*> stats stop
StatisticsID: Ind0xffffff08b9536188
lun:ed95d953-1560-4f74-9006-85b352f58fcd:instance_uuid:ed95d953-1560-4f74-9006-85b352f58fcd
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.0:186%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.1:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.2:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.3:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.4:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.5:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.6:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.7:0%
```

Misaligned

The following example shows misaligned I/O.

1. Create a partition that does not align to a 4KB boundary. This is not default behavior on modern OSs.

```
[root@jfs0 iscsi]# fdisk -u /dev/sdb
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First sector (32-20971519, default 32): 33
Last sector, +sectors or +size{K,M,G} (33-20971519, default 20971519):
Using default value 20971519

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

2. The partition has been created with a 33-sector offset instead of the default 32. Repeat the procedure outlined in the section “Aligned.” The histogram appears as follows:

```
FAS8040SAP-02*> stats stop
StatisticsID: Ind0xffffff0468242e78
lun:ed95d953-1560-4f74-9006-85b352f58fcd:instance_uuid:ed95d953-1560-4f74-9006-85b352f58fcd
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.0:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.1:136%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.2:4%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.3:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.4:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.5:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.6:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.7:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_partial_blocks:31%
```

The misalignment is clear. The I/O mostly falls into the `.1` bucket, which matches the expected offset. When the partition was created, it was moved 512 bytes further into the device than the optimized default, which means that the histogram is offset by 512 bytes.

Additionally, the `read_partial_blocks` statistic is nonzero, which means I/O was performed that did not fill up an entire 4KB block.

Redo Logging

The procedures explained here are applicable to datafiles. Oracle redo logs and archive logs have different I/O patterns. For example, redo logging is a circular overwrite of a single file. If the default 512-byte block size is used, the write statistics look something like this:

```
FAS8040SAP-02*> stats stop
StatisticsID: Ind0xffffff0468242e78
lun:ed95d953-1560-4f74-9006-85b352f58fcd:instance_uuid:ed95d953-1560-4f74-9006-85b352f58fcd
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.0:12%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.1:8%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.2:4%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.3:10%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.4:13%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.5:6%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.6:8%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.7:10%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_partial_blocks:85%
```

The I/O would be distributed across all histogram buckets, but this is not a performance concern. Extremely high redo-logging rates might, however, benefit from the use of a 4KB block size. In this case,

it is desirable to make sure that the redo-logging LUNs are properly aligned. However, this is not as critical to good performance as datafile alignment.

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.

TR-3633-0717