An Oracle Technical White Paper
January 2014

Understanding the Use of Fibre Channel in the Oracle ZFS Storage Appliance

ORACLE

Executive Overview

The Oracle ZFS Storage Appliance products offer Fibre Channel (FC) target LUN functionality that enables users to configure volumes in the appliance's storage pool(s) and serve them out as Fibre Channel block devices. Properly configuring these LUNs is key to maximizing their performance and efficiency. This paper provides instructions, recommendations, and examples for how to realize this FC LUN functionality, including:

- Instructions for setting up a Fibre Channel configuration using an Oracle ZFS Storage Appliance two-node cluster storage subsystem.
- Recommendations for designing redundant I/O paths in a Storage Area Network (SAN), using SAN zoning, and creating partitions and file systems for various operating systems.
- Guidelines, recommendations and examples for aligning partitions with the block size of created FC LUNs – a critical function.
- Lastly, recommendations on the use of solid state drive (SSD) cache devices, based on Online Transaction Processing (OLTP) performance testing conducted on various configurations.

Introduction

Along with its NAS-based file sharing capabilities, the Oracle ZFS Storage Appliance can serve out block-type devices using either Fibre Channel or iSCSI. The support of Fibre Channel LUN devices enables the Oracle ZFS Storage Appliance to function in a customer SAN environment as a Fibre Channel block storage device.

The Fibre Channel target mode supports multiple I/O paths between host(s) and a target device on the Oracle ZFS Storage Appliance. Properly developing the architecture for either a dual- or single-fabric SAN for an Oracle ZFS Storage Appliance configuration means ensuring that path redundancy is established.

The next step in the configuration process is to map customer and application requirements for space, reliability, performance and application I/O profile to volume/LUN size requirements, pool profile choice, and the choice of SSD type cache devices. This paper addresses best design practices and performance-related guidelines and examples for this process.

Equally important is creating partitions on the configured LUNs in such a way to ensure optimal performance results. Each operating system uses its own type of tools, using default values for start locations of partitions that are often not optimal. Partition tools for each major type of operating system are reviewed, followed by guidelines on how to achieve an optimal match between a partition’s start location and the ZFS block size used on a LUN.

Using the Oracle ZFS Storage Appliance with FC LUNs

The following overview presents the Fibre Channel architecture available with the Oracle ZFS Storage Appliance, how that architecture functions, and the overall setup process for utilizing FC LUNs on the system.

Fibre Channel Architecture of the Oracle ZFS Storage Appliance

The supported Fibre Channel Host Bus Adapters (HBAs) for the Oracle ZFS Storage Appliance all have two ports. This means that each unit provides two connections to a Fibre Channel SAN infrastructure. Using a dual-fabric-based SAN design ensures full path redundancy to the host.

The configuration described in this paper details an Oracle ZFS Storage dual-node cluster using a dual-fabric SAN infrastructure as shown in the following diagram.

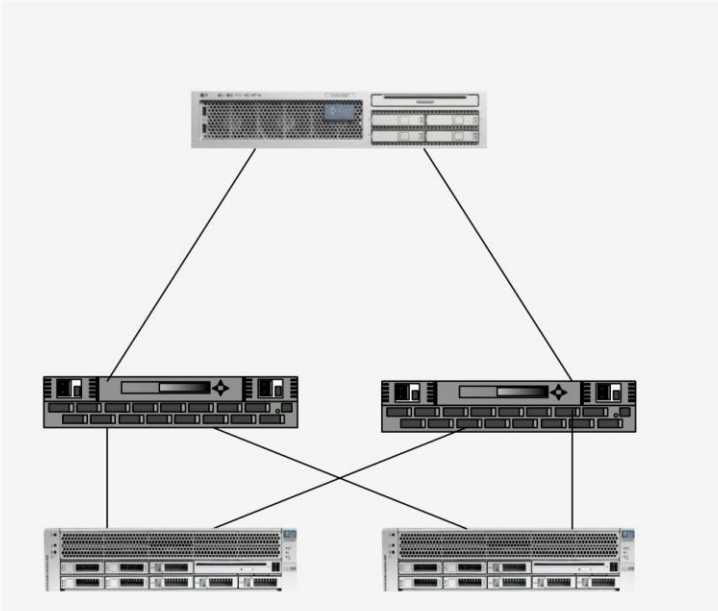


Figure 1. SAN connection diagram

To ensure no single point of failures in the SAN components/paths, both host HBA port connections always 'see' both Oracle ZFS Storage nodes. For each FC LUN on the Oracle ZFS Storage Appliance, each host HBA path sees both an active connection and a standby connection.

Theory of Operation

Each Oracle ZFS Storage node runs an instance of an embedded Oracle Solaris operating system. The multipath element to handle I/O redirection from failed I/O paths to remaining active path is part of the Common Multi-protocol SCSI Target (COMSTAR) framework.

All configuration information within the COMSTAR framework, like target and initiator group(s) information, is kept in sync between the two Oracle ZFS Storage nodes by the cluster functionality in the nodes.

LUNs configured in a pool on the Oracle ZFS Storage Appliance appear on FC ports on both nodes of the Oracle ZFS Storage cluster. The two paths to the node that owns the pool are visible to the host as A*ctive* paths, while the path to the LUN through the node that does not own the pool has *Standby* status.

This means a host can initiate a failover of the traffic between active paths to a LUN, like a dual FC connection to an Oracle ZFS Storage node.

A failover between an Active and Standby path can be initiated by an Oracle ZFS Storage node failover. All pools must fail over to the node that had the Standby path status before the node failover.

Fibre Channel Setup and Configuration Sequence

In order to make the Fibre Channel installation and configuration process as easy as possible, execute the following steps in the listed sequence. More details for these steps are provided in following sections.

- Oracle ZFS Storage Appliance hardware preparation
    1. Add an SG-PCIE2FC-QF4 (4 Gb) or SG-XPCIE2FC-QF8-Z (8 Gb) dual-ported FC HBA card in each node in the recommended slot positions for FC HBAs.
    2. Power up the nodes and verify the HBAs have been recognized.
    3. Set the Oracle ZFS Storage HBAs up for target mode and identify the World Wide Names (WWNs) of the FC ports. These are needed when setting up the zones in the SAN switch(es).
- Fibre Channel host connections
    1. Verify the FC HBA(s) on the host(s) are recognized by the operating system (OS) on the host. In this paper, an Oracle Solaris host is used with one dual-port FC HBA.
    2. Identify the WWNs of the host HBA FC ports.
- SAN configuration and zoning setup
    1. Make the correct cable connections among the switch(es), host and Oracle ZFS Storage cluster. Verify all connection lights come up, indicating an established link with the switch at the desired speed.
    2. Configure zoning using HBA port and Oracle ZFS Storage FC HBA port WWNs.
    3. Configure the FC target and initiator groups on Oracle ZFS Storage nodes.
       In order to define access to the Oracle ZFS Storage nodes using the WWNs of each of the host HBA ports, target and initiator groups must be configured on the Oracle ZFS Storage nodes.

- LUN creation and configuration on Oracle ZFS Storage nodes
    1. Create LUNs and set up the membership of target and initiator groups to the LUNs.
    2. Verify that the LUN attributes, like block size and cache behavior requirements on the Oracle ZFS Storage node, are set as required. Attributes can be changed at any time. Note, however, that once you have created a LUN, its block size attribute cannot be changed. Carefully consider the block size value before setting it.
    3. Verify the visibility of configured LUNs on the host(s).
    4. Detect the GUID (Global unique identifier) in the device name of the LUN by matching it with the GUID presented by the Oracle ZFS Storage Appliance BUI.
    5. Partition and label the LUN using tools of the host OS, like fdisk, format, or parted.
    6. Ensure proper block alignment between the start of a partition and the file system to be used relative to the start of the raw LUN device. See the section titled "Partition Alignment" for more details.
    7. Verify the SAN zoning configuration.

This paper describes the commands you can use on an Oracle Solaris host to verify that access to the configured LUNs is established. For information on setting up VMware to recognize the asymmetric logical unit access (ALUA) capabilities of the Oracle ZFS Storage Appliance, see Appendix B, "VMWare ALUA Support Setup."

Also see the Oracle ZFS Storage Appliance Online Help manual, section "Configuring FC Client Multipathing."

Note: Use root privileges on Oracle Solaris to ensure proper permissions exist to execute the commands discussed later in this document.

Setup Processes for Utilizing Fibre Channel LUNs

The following provides more detail on the FC LUN setup and configuration.

Preparing Oracle ZFS Storage Hardware

First, add the FC HBA cards in each Oracle ZFS Storage node. When working on a cluster configuration, you must first fail all services over to the remaining node and power down the node to which you will add the FC card.

Make sure the FC HBA, SG-XPCIE2FC-QF8-Z (8 Gb), is placed in the correct slot as documented in the Oracle ZFS Storage Appliance's customer service manual. The References section at the end of this document provides a link to the documentation set for the Oracle ZFS Storage Appliance.

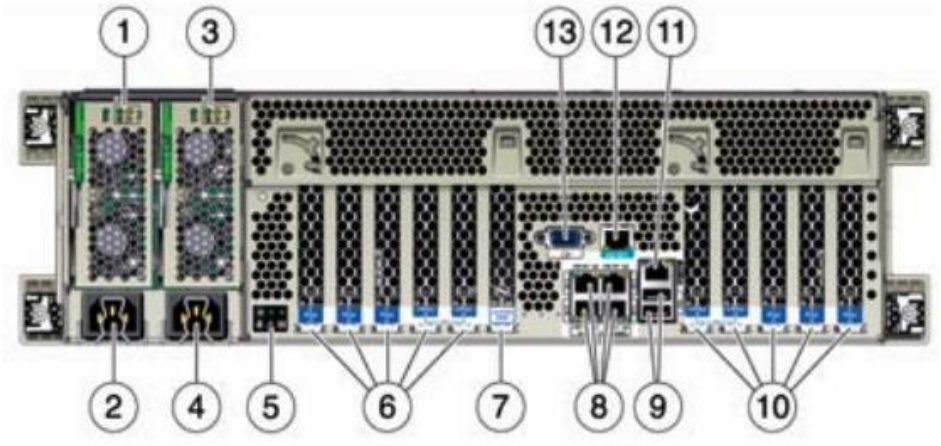


Figure 2. PCIe slot numbering at 6 and 10 on Oracle ZFS Storage Appliance

PCI slot numbering is shown on the back of the Oracle ZFS Storage unit, items 6 and 10.

After system powerup, use the hardware view in the GUI to verify if the FC card has been recognized and is located in the PCIe slot as planned.

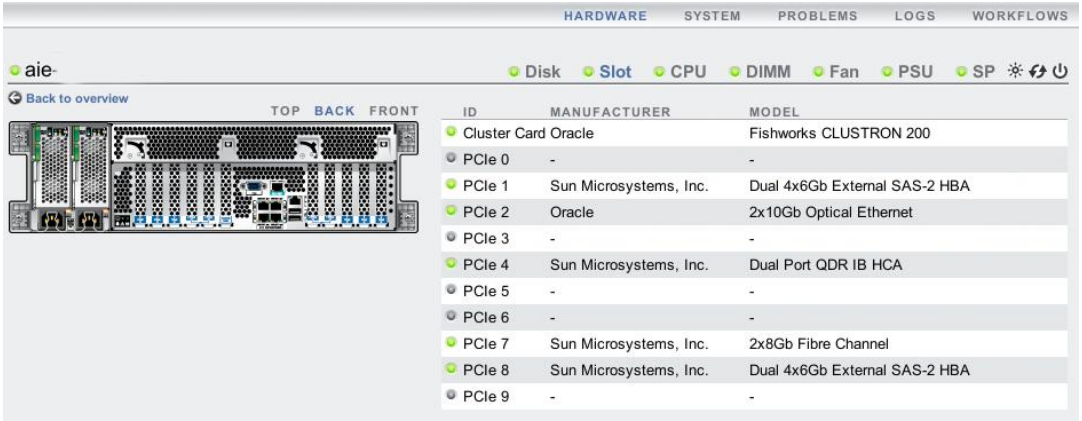


Figure 3. HBA verification in Oracle ZFS Storage Appliance BUI

To see if the HBAs are receiving an active connection from the fabric, check for the lighted (red) dots in the port icon as shown in the Configuration<SAN BUI window:

Figure 4. Active links in Ports

Verifying Fibre Channel Host Connections

Next, you will verify the visibility of the configured PCIe cards (FC HBAs) to the host's operating system; in this case, Oracle Solaris. After connecting the HBA FC ports to the SAN infrastructure, you must establish the correct functioning of the host FC HBAs and identify the WWNs of the HBA ports. For Oracle Solaris, use the command `cfgadm`.

Verify that the HBA FC ports are available using the `cfgadm -al` command. The following code example shows the output for the HBAs.

```
bash-3.2# cfgadm -al
Ap_Id                      Type             Receptacle   Occupant         Condition
PCI0                       scsi/hp          connected    configured       ok
PCI1                       etherne/hp       connected    configured       ok
c14                        fc-fabric        connected    unconfigured   unknown
c15                        fc-fabric        connected    unconfigured   unknown
```

Note: Non-relevant lines of information are removed in the preceding code example.

To identify the WWNs of the HBA Fibre Channel ports, use the `fcinfo` command as follows.

```
bash-3.2# fcinfo hba-port
HHBA Port WWN: 210000e08b85b57b
        Port Mode: Initiator
        Port ID: 10800
        OS Device Name: /dev/cfg/c14
        Manufacturer: QLogic Corp.
        Model: 371-4522-02
        Firmware Version: 05.04.03
        FCode/BIOS Version:  BIOS: 2.10; fcode: 3.06; EFI: 2.04;
        Serial Number: 0402L00-1047843144
        Driver Name: qlc
        Driver Version: 20110321-3.05
        Type: N-port
        State: online
        Supported Speeds: 2Gb 4Gb 8Gb
        Current Speed: 8Gb
        Node WWN: 20000024ff24952e
        Max NPIV Ports: 254
        NPIV port list:


HBA Port WWN: 210100e08b85b57b
        Port Mode: Initiator
        Port ID: 10900
        OS Device Name: /dev/cfg/c15
        Manufacturer: QLogic Corp.
        Model: 371-4522-02
        Firmware Version: 05.04.03
        FCode/BIOS Version:  BIOS: 2.10; fcode: 3.06; EFI: 2.04;
        Serial Number: 0402L00-1047843144
        Driver Name: qlc
        Driver Version: 20110321-3.05
        Type: N-port
        State: online
        Supported Speeds: 2Gb 4Gb 8Gb
        Current Speed: 8Gb
        Node WWN: 20000024ff24952f
        Max NPIV Ports: 254
        NPIV port list
```

From the preceding output, you can determine both the WWN of the host bus adapter and the AP-ID (C14/C15). Since the AP-ID is used as the identifier in the device path, it is good practice to use it as part of the initiator name in the initiator group setup on the Oracle ZFS Storage Appliance.

SAN Configuration and Zoning Setup

Now that both the initiator and target FC-HBAs have been configured, proper routing in the SAN must be defined by means of SAN zones. To ensure HBAs will not see each other, a minimum of one zone per initiator per fabric is needed. In the following examples, a zone per logical connection is used.

Before creating zones, make sure the SAN configuration is correctly cabled. The following SAN configuration ensures optimal availability. The colored lines show the logical 'zoned' connections.

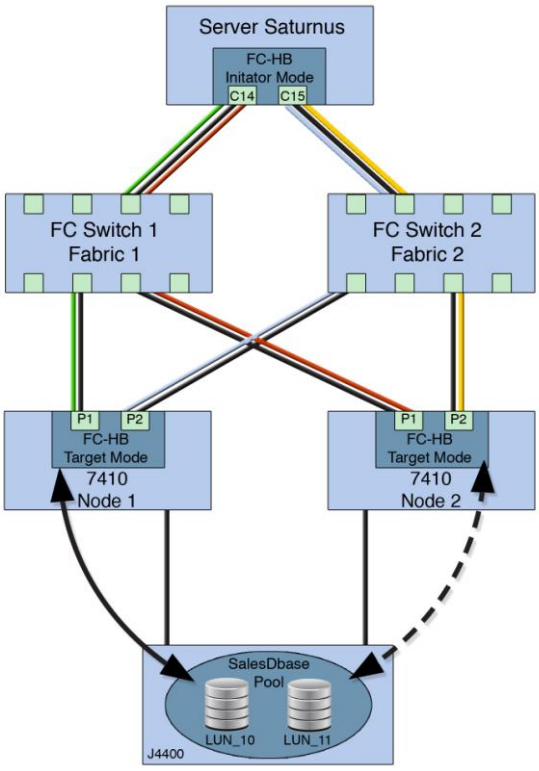


Figure 5. SAN wiring diagram

Make sure you have two paths to each storage controller (Oracle ZFS Storage node). This redundancy avoids outages if a SAN connection between switch and storage controller fails – a situation that would *not* initiate an automatic node failover.

Different methodologies of zoning exist, like WWN-based zoning and SAN switch port-based zoning. The preferred methodology often depends on a company's IT security policies and configuration management rules.

In the examples, the port WWN zoning methodology is used. Give the WWNs in the switch a logical name using the alias option. In this example a combination of host name and AP-IDs are used in aliases for host WWNs and Oracle ZFS Storage node name and port name for aliases for Oracle ZFS Storage WWNs.

The following example shows that four zones are used.

TABLE 1. SAN ZONING SETUP

ZONE NAME	MEMBER 1		MEMBER 2	
ZONE A	Host1-HBA-C14	21:00:00:e0:8b:85:b5:7b	ZFSSA-NODE-1 Port 1	21:00:00:24:ff:31:82:66
ZONE B	Host1-HBA-C14	21:00:00:e0:8b:85:b5:7b	ZFSSA-NODE-2 Port 1	21:00:00:24:ff:31:83:04
ZONE C	Host1-HBA-C15	21:10:00:e0:8b:85:b5:7b	ZFSSA-NODE-1 Port 2	21:00:00:24:ff:31:82:67
ZONE D	Host1-HBA-C15	21:01:00:e0:8b:85:b5:7b	ZFSSA-NODE-2 Port 2	21:00:00:24:ff:31:83:05

This zone setup creates four logical paths between the host and the Oracle ZFS Storage cluster, with two paths per node. With the zones properly set up, you can now configure FC target and initiator groups on the Oracle ZFS Storage Appliance.

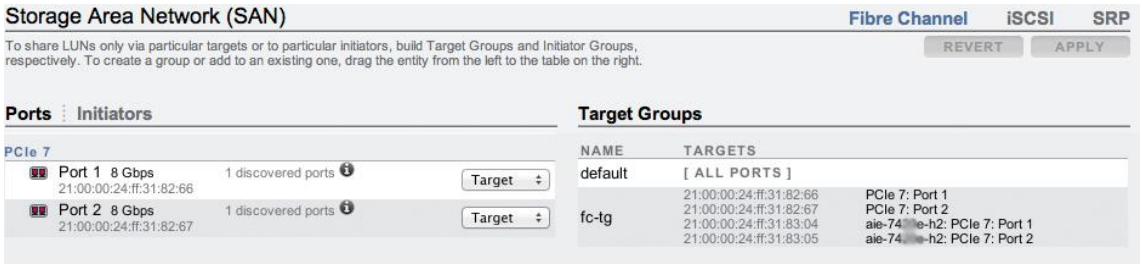
The construct of target groups is used within the Oracle ZFS Storage Appliance's interface to allocate LUNs to certain FC ports of target FC HBAs in the Oracle ZFS Storage Appliance. A target group acts as a pool of target FC ports from which LUNs are made visible to the outside world.

Initiator groups in the Oracle ZFS Storage Appliance are used as a means to control host (initiators) access to the LUNs. So an initiator group acts as a pool of initiator FC ports that can be given access to a LUN in the Oracle ZFS Storage Appliance.

Start by setting the Fibre Channel ports to be used in the Oracle ZFS Storage Appliance to Target mode, as seen in the pull-down menu selection in the following figure.

In order to have an active failover path within the Oracle ZFS Storage node and a node service failover path to the other node, all four target ports of the Oracle ZFS Storage cluster need to be configured in the same target group as shown in the following screenshot. Configure one node, then move over to the other node and add the two target FC ports from that node to the already created target group on the first node. This step is a drag-and-drop action on the screen.

Figure 6. FC target group setup



Next, host ports are allocated in an initiator group. The initiator group is used later in the setup process, when LUNs are created. Provide logical, meaningful names for the initiator ports.

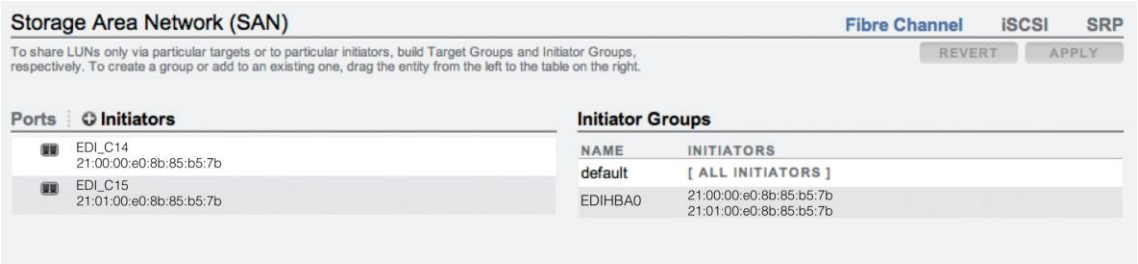


Figure 7. FC Initiators setup

As of major firmware release 2013 of the Oracle ZFS Storage Appliance, FC initiators can be made members of more than one initiator group and LUNs can be made members of multiple initiator groups. This capability allows creation of a dedicated initiator group for backup and another containing the initiators for production data access. A LUN can now be made a member of both initiator groups.

LUN Creation and Configuration on Oracle ZFS Storage Nodes

Select LUNs and use the + sign to add new LUNs.

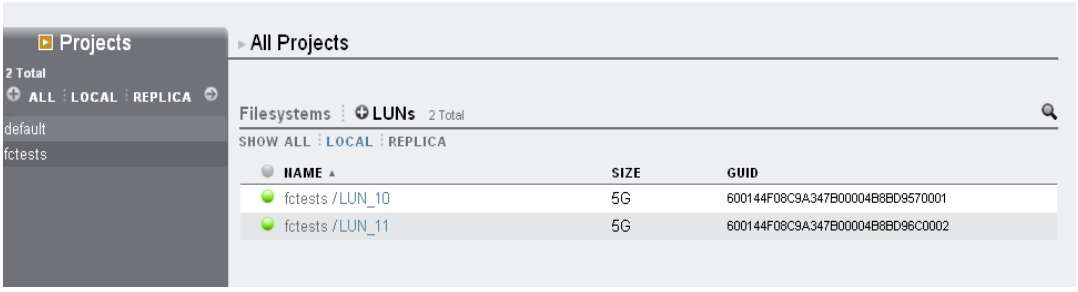


Figure 8. Adding LUNs to Node A

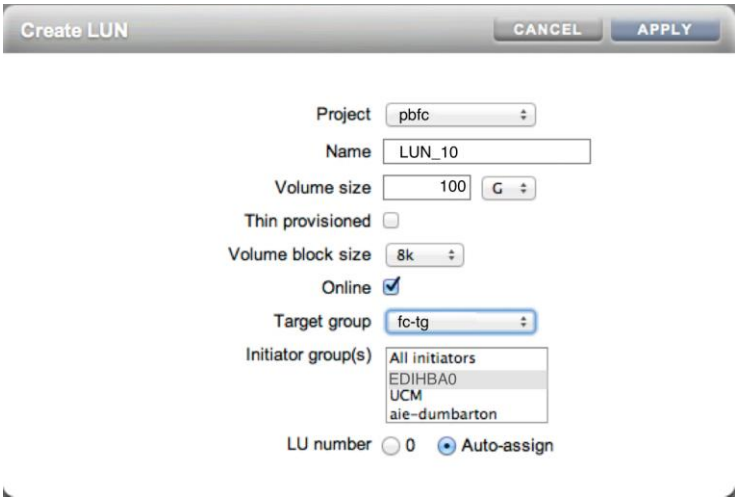


Figure 9. Setting FC LUN specifications

Specify the volume size for the LUN you are creating. Select the block size carefully; its setting depends on the type of I/O access patterns used by the applications that will use the LUN. A block size that matches the natural database blocksize is recommended for OLTP-type applications. For databases that use multiple block sizes, either use the smallest database block size for record size or use multiple LUNS, each with different block sizes. Redo logs (being streaming workload) should use 128K record sizes. Larger block sizes (up to 128K) are appropriate for streaming I/O-type applications. Once a LUN is created, these settings cannot be changed.

Figure 10. LUN Initiator and Target group settings

Select to which target group and initiator group(s) to allocate the LUN(s).

You can specify various LUN properties in the BUI General section, under the Share (LUN) properties section. Each of these listed properties can be changed at any time to fine-tune the LUN's performance characteristics.

Figure 11. LUN Properties configuration

Verify the configuration on the host

Verify that the HBAs are configured using the `cfgadm -al` command. If they are not configured, use the command `cfgadm -c configure <Ap-ID>`.

The following shows the command output for the HBAs.

```
bash-3.2# cfgadm -al
Ap_Id                          Type           Receptacle    Occupant     Condition
PCI0                           scsi/hp        connected     configured    ok
PCI1                           ethernet/hp    connected     configured    ok
c14                            fc-fabric      connected     configured   unknown
c14::21000024ff318266          disk           connected     configured   unknown
c14::21000024ff318304          disk           connected     configured   unknown
c15                            fc-fabric      connected     configured   unknown
c15::21000024ff318267          disk           connected     configured   unknown
c15::21000024ff318305          disk           connected     configured   unknown
```

The preceding output shows that the zoning is properly set up. C14 has access to port 1 of both Oracle ZFS Storage nodes and C15 has access to port 2 of both nodes. For complete path verification, use the `luxadm` command as shown further in this section.

Partition and label the LUNs

Use the `format` command to check that the configured LUNs are seen by the host.

```
bash-3.2# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c16t500000E010CC3B30d0 <SUN146G cyl 14087 alt 2 hd 24 sec 848>
          /scsi_vhci/ssd@g500000e010cc3b30
       1. c16t500000E010CCD120d0 <SUN146G cyl 14087 alt 2 hd 24 sec 848>
          /scsi_vhci/ssd@g500000e010ccd120
       2. c16t600144F0C0ACA00400004B8C13B60001d0 <SUN-SunStorage7410-1.0 cyl 160 alt 2
    hd 254 sec 254>
          /scsi_vhci/ssd@g600144f0c0aca00400004b8c13b60001
```

```
       3. c16t600144F0C0ACA00400004B8C13CC0002d0 <SUN-SunStorage7410-1.0 cyl 160 alt 2
     hd 254 sec 254>
          /scsi_vhci/ssd@g600144f0c0aca00400004b8c13cc0002
       4. c16t600144F08C9A347B00004B8BD96C0002d0 <SUN-SunStorage7410-1.0 cyl 160 alt 2
     hd 254 sec 254>
          /scsi_vhci/ssd@g600144f08c9a347b00004b8bd96c0002
       5. c16t600144F08C9A347B00004B8BD9570001d0 <SUN-SunStorage7410-1.0 cyl 160 alt 2
     hd 254 sec 254>
          /scsi_vhci/ssd@g600144f08c9a347b00004b8bd9570001
Specify disk (enter its number):
```

To use the LUNS, you must label them. The `label` command creates a partition table on the LUN. For further information on the type of partition to use and block alignment considerations, see the topic "Partition Alignment" in the section "Design Best Practices."

Verify the SAN zoning configuration

Next, verify that the SAN zoning and cabling are correctly set up. Four paths should be visible to the LUN: two active, to the Oracle ZFS Storage node that actively serves the LUN, and two standby**,** to the other Oracle ZFS Storage node.

Use the `luxadm probe` command to find the available paths:

```
bash-3.2# luxadm probe
No Network Array enclosures found in /dev/es
Found Fibre Channel device(s):
  Node WWN:500000e010cc3b30  Device Type:Disk device
    Logical Path:/dev/rdsk/c16t500000E010CC3B30d0s2
  Node WWN:500000e010ccd120  Device Type:Disk device
    Logical Path:/dev/rdsk/c16t500000E010CCD120d0s2
  Node WWN:2000001b32135c63  Device Type:Disk device
    Logical Path:/dev/rdsk/c16t600144F0C0ACA00400004B8C13B60001d0s2
  Node WWN:2000001b32135c63  Device Type:Disk device
    Logical Path:/dev/rdsk/c16t600144F0C0ACA00400004B8C13CC0002d0s2
  Node WWN:20000024ff318266  Device Type:Disk device
    Logical Path:/dev/rdsk/c16t600144F08C9A347B00004B8BD96C0002d0s2
  Node WWN:20000024ff318266  Device Type:Disk device
    Logical Path:/dev/rdsk/c16t600144F08C9A347B00004B8BD9570001d0s2
```

The devices from WWN `20000024ff318266` (the WWN of the HBA of the Oracle ZFS Storage node) are the LUNs of interest.

Only the info for one of the LUNs is shown in the following `luxadm display` command output. The info for the second LUN should be identical. The output shows that the LUN has active connections to the ports of the Oracle ZFS Storage node that serves the LUN (Device Address `20000024ff318266,1`). The two connections to the other node are in standby state.

```
bash-3.2# luxadm display 20000024ff318266
DEVICE PROPERTIES for disk: 2001001b322be2b4
  Vendor:               SUN
  Product ID:           Sun Storage 74X0
  Revision:             1.0
  Serial Num:
  Unformatted capacity: 5120.000 Mbytes
  Read Cache:           Enabled
    Minimum prefetch:   0x0
    Maximum prefetch:   0x0
```

```
  Device Type:          Disk device
  Path(s):
  /dev/rdsk/c16t600144F08C9A347B00004B8BD96C0002d0s2
  /devices/scsi_vhci/ssd@g600144f08c9a347b00004b8bd96c0002:c,raw


Controller           /devices/pci@9,600000/SUNW,qlc@2,1/fp@0,0
    Device Address              21000024ff318266,1
    Host controller port WWN    210100e08ba5b57b
    Class                       primary
    State                       ONLINE




Controller           /devices/pci@9,600000/SUNW,qlc@2,1/fp@0,0
    Device Address              21000024ff318305,1
    Host controller port WWN    210100e08ba5b57b
    Class                       secondary
    State                       STANDBY


Controller           /devices/pci@9,600000/SUNW,qlc@2/fp@0,0
    Device Address              21000024ff318267,1
    Host controller port WWN    210000e08b85b57b
    Class                       primary
    State                       ONLINE


Controller           /devices/pci@9,600000/SUNW,qlc@2/fp@0,0
    Device Address              21000024ff318304,1
    Host controller port WWN    210000e08b85b57b
    Class                       secondary
    State                       STANDBY
```

Write Cache Considerations

The Oracle ZFS Storage Appliance uses extensive data caching based on the functionality the ZFS file system offers.

ZFS uses a two-tiered cache system: Adaptive Replacement Cache (ARC) in main memory and a second-level cache using solid state drives (SSDs). This second-level cache is split into read and write components, the Second Level Adjustment Replacement Cache (L2ARC) and the ZFS Intend log (ZIL).

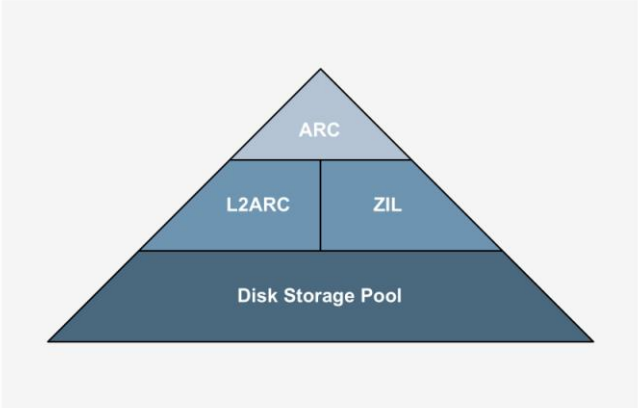


Figure 12. ZFS cache model

The use of SSDs is optional in a ZFS pool. When no SSDs are used, blocks for the ZIL are kept on disks in the ZFS pool and L2ARC functionality is not available.

The function of the L2ARC read cache is to reduce the latency for reads between the ARC and disks when no blocks are found in the ARC. Since the L2ARC is large compared to the ARC, it takes some time for it to become populated, or warmed up. Random read workloads benefit most from L2ARC.

ZFS ZIL operations are always a part of a Data Management Unit (DMU) transaction. When a DMU transaction is opened, an associated ZIL transaction also opens. These transactions accumulate in memory until an fsync or O_DSYNC write occurs, when the transactions are committed to stable storage. The associated ZIL transaction is, in most cases, discarded when the DMU transaction commits.

Note: A good source for more in-depth information on the functioning of the ZFS cache mechanism can be found in various BLOGs from ZFS developers, listed in "References," Appendix C.

When creating LUNs on the Oracle ZFS Storage Appliance, you are given the option to use the appliance main memory for caching for performance or to keep all writes on stable storage for guaranteed data consistency.

When write performance is critical, configure log devices (Write Optimized SSDs) and, to ensure data consistency, do not enable the write cache for the LUN.

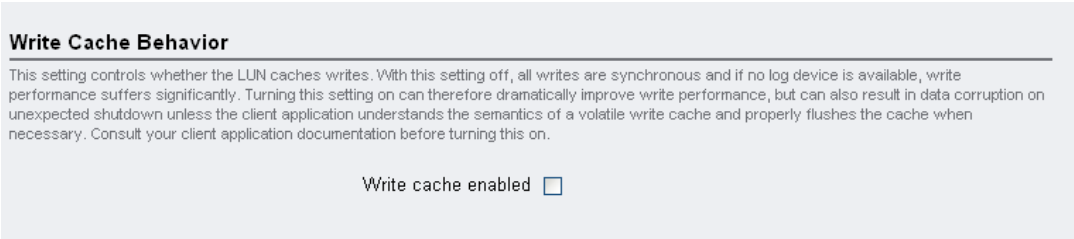


Figure 13. Write cache behavior

For each LUN, ZIL behavior can be influenced by setting the synchronize write cache bias property to either latency optimized or throughput optimized. These settings enable control over which LUNs use the log devices (write optimized SSDs) in the pool the LUNs are part of.

Latency optimized: This setting commits writes immediately to write-optimized SSDs to reduce latency. By immediately committing writes to the SSDs, writes are protected against failure. This setting makes heavy use of write-optimized SSDs and thus is limited by the devices' performance. More write-optimized SSDs striped together provide better I/O operations per second (IOPs) and bandwidth performance.

Throughput optimized: This means that writes bypass the write SSD accelerator and commit directly to disks. This setting leads to higher transaction latency for a single transaction and only provides reasonable throughput when highly multithreaded writes are hitting a LUN. Since only a few workloads could provide such highly parallel load, you should avoid this setting in order to maintain good performance levels.

However, this log bias setting can be used to store Oracle data files accessed by database writers. The setting allows the write bias SSD to service other, more critical I/O such as the redo log with lower latency. During administrative tasks, which are often single threaded, a best practice is to toggle the setting to avoid large slowdowns.

Testing the Configuration for Failure and Recovery Scenarios

To test the reaction to a number of link failures, a simple load has been set up on the host, creating one read stream from a single LUN as shown in the following analytics screenshot.

Figure 14. Load under normal situation

This output shows that the load is distributed evenly over C14 and C15, and I/O requests come in on port 1 and port 2 of one Oracle ZFS Storage node b.

Single Connection Failure Between Fabric and Host HBA Port

The next step is to fail the connection from C14 into the fabric. The next screenshot shows that traffic continues without interruption over the remaining connection to the FC LUN. Restoring the link restores the dual path access to the LUN to normal.

Figure 15. One FC Link failure

Single Connection Failure Between Fabric and Oracle ZFS Storage Appliance Node

The next step is to fail a link between an active port of the Oracle ZFS Storage node and the fabric (16:13). This time there is a small disruption in the transfer of data, and the data transfer from the remaining path resumes at 16:13:21.

Figure 16. Link disconnected from active Oracle ZFS Storage node

When the link is restored (at 16:13:40), the data flow is directly redistributed over both active ports on the Oracle ZFS Storage node.

Failure of Links to Both Active Ports of Oracle ZFS Storage Node

This is a double failure scenario. Failing both links to the Oracle ZFS Storage Appliance with paths to the FC LUN results in halting the I/O to the LUN until the links are re-established. Failover of the data traffic to the node with the standby path will occur when a node failover is initiated.

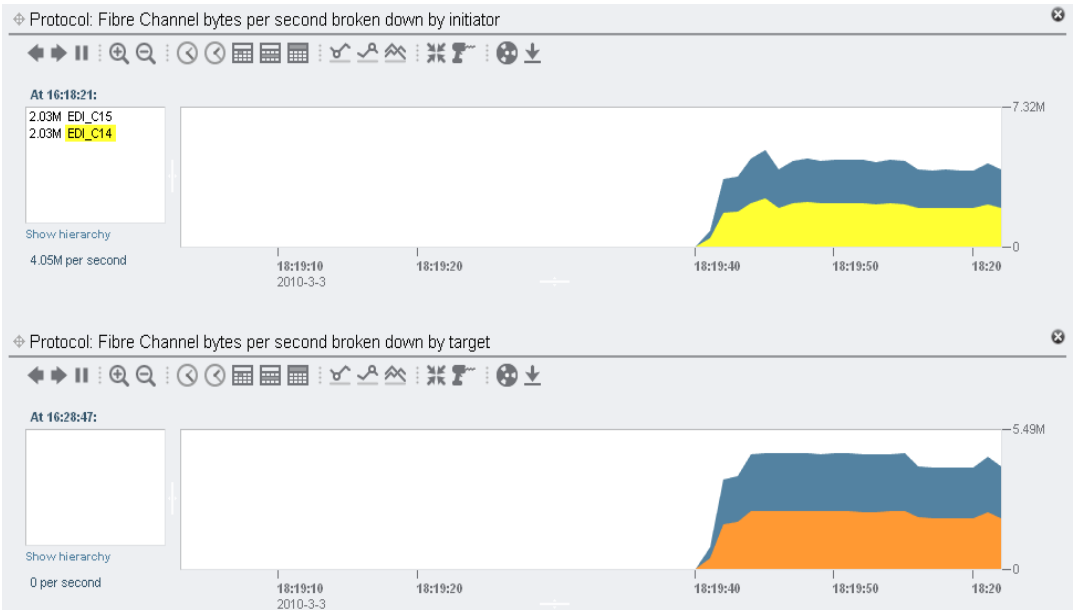


Figure 17. Oracle ZFS Storage node failover

Oracle ZFS Storage Node Failure

Triggering a node takeover from a node that is actively serving FC LUNs results in the I/O of that LUN being taken over by the requesting node. The failover for a single LUN takes about 30 seconds.

Summary of Failure Scenarios

TEST TYPE	RESULT
Failure of link from one host port to fabric	Active path between remaining host HBA port and active port on Oracle ZFS Storage Appliance node continue to serve I/O without any disruption.
Failure of one of the links of active ports from Oracle ZFS Storage Appliance node	I/O is temporary suspended for about 25 seconds before I/O resumes on the remaining active path. When path is restored, I/O is picked up instantly
Double failure scenario; failure of both links to active ports on Oracle ZFS Storage Appliance node	I/O stops. A cluster node failover must be initiated for traffic to resume on the node with the standby paths.

TEST TYPE	RESULT
User-initiated node service takeover	I/O temporarily suspends for about 30 seconds. Then I/O is taken over by the other Oracle ZFS Storage Appliance node.

Design Best Practices

Host Connection Considerations

An important design aspect in an FC SAN architecture is the number of hosts connected to a storage subsystem and whether the hosts will share ports and/or LUNs on the storage side of the SAN.

Sharing LUNs between hosts is typically done in a host cluster type configuration or in high performance computing (HPC) environments such as the Lustre file system or Oracle Automatic Storage Management (ASM) storage. When a LUN is actively shared among multiple hosts, the software on those hosts is responsible for resolving possible data access contention that, if not properly handled, might result in data corruption. Often the use of a file system that has sharing capabilities is required. Oracle's QFS file system is an example of a file system that can function in such environments. Note, however, that describing these types of solutions is outside the scope of this paper.

When dealing with architectures in which multiple hosts connect to the same storage subsystem ports, you must carefully consider the sizing of the SCSI queue mechanism on each host. Proper sizing prevents the storage subsystem SCSI queue from getting overloaded with multiple SCSI commands from the different hosts.

Setting Command Queue Depth

A host can issue multiple I/O commands at a time to a LUN through the command tag queuing mechanism. The number of commands in the queue varies per vendor operating system and the driver implementation for a specific FC HBA.

The Oracle ZFS Storage FC target driver can handle up to 2048 commands in the queue for an HBA. From this characteristic, the maximum queue depth per LUN must be derived for each host HBA port present in the architecture. To avoid negative impacts to performance because of SCSI timeouts, you must prevent overruns of the target HBA port queue for all types of situations.

In a simple configuration with one LUN and one host FC connection, the maximum queue depth on the host cannot be set higher than 2048. If this LUN is shared by *n* hosts, then for each host the queue depth must be divided by the number of hosts sharing the LUN.

When configuring multiple LUNs on an Oracle ZFS Storage Appliance, the queue depth per LUN on the host must be set to 2048 divided by the number of LUNs. If those LUNs are going to be shared

among multiple hosts, the number has to be further divided by the number of hosts: 2048/*n*(LUNs) * *n*(hosts), then rounded off to the nearest lower integer number.

Lastly, consider whether the Oracle ZFS Storage Appliance will be used in an active/active cluster configuration. When one of the nodes fails, the corresponding FC port on the remaining node serves the LUNs that are configured on its counterpart. To be safe, use all LUNs on both cluster nodes' corresponding HBA parts for the queue depth calculation.

The equation is: 2048/(*n* **l*) or 2048/(*n* **l* * 2) for the Oracle ZFS Storage cluster,
where *l* is the number of LUNs on an Oracle ZFS Storage target port and *n* the number of hosts sharing the LUNs.

Do this calculation for each Oracle ZFS Storage target port.

To reiterate, carefully plan the amount of LUNs and hosts that connect to the target ports to avoid overloading the command queues. In the rare case that several hosts share LUNs, further reduction is required. Consider all ports in a failover configuration to participate on the load.

Values cannot be set loosely. For each host for which changes have been made, you must reboot. Use conservative values, choosing the smaller number rather than the higher. New LUNs could be added later and queue depth values on all the hosts must be changed again. Each vendor operating system has a different configuration mechanism for setting the host maximum queue depth.

Setting Command Queue Depth in Oracle Solaris

For Oracle Solaris, two global variables, `sd_max_throttle` and `ssd_max_throttle`, set queue depth. Which one is used for a specific HBA depends on whether the HBA's driver binds itself to either the sd or ssd driver. In both cases, the variable controls the queue depth used per target LUN. The default setting for `s(s)d_max_throttle` is 256. This means that when using more than eight LUNs per Oracle ZFS Storage HBA port, the value of `s(s)d_max_throttle` must be lowered.

Global Method for Setting `s(s)d_max_throttle` in the Oracle Solaris Kernel

To set `s(s)d_max_throttle`, add the following line to the kernel file, `/etc/system`:

```
set ssd:ssd_max_throttle=x
```

or

```
set sd:sd_max_throttle=x
```

where *x* is the maximum queue depth per LUN as calculated following the previously described rules.

A system reboot is required to make the kernel use the newly configured queue depth.

For further information, see the help file in the Oracle ZFS Storage Appliance system interface, https://<*your.ip.address*>:215/wiki/index.php/Configuration:SAN:FC#Queue_Overruns.

Partition Alignment

The ZFS volumes use a fixed block size to store data in volumes. The size of the blocks can be specified when creating a volume or a LUN. Ideally, each I/O request from a client uses the same block size and each of those blocks start at the same point as the corresponding block in the ZFS volume – they are aligned. If a read from a client is the same size but not aligned, the ZFS must access at least two blocks in order to return the data to the client. Moreover, for writes which are naturally aligned, having a partition misalignment will cause those writes to issue reads which are otherwise not necessary, causing the so-called Read-Modify-Write penalty.

This misalignment results in disk I/O overhead and reduces LUN performance from a client's perspective. This inefficiency is particularly important in a random I/O OLTP type environment.

What can cause misalignment?

In order for a client to access a LUN, a partition table has to be written on the LUN. Partitioning implementations differ, depending on operating systems. Partition schemes are still based on physical disk geometry parameters, sectors, tracks, cylinders and heads. For LUNs from storage subsystems, those parameters are purely virtual and do not have any correlation with the disks used in the storage subsystem.

So in this context, it is important that the start sector of a partition created on a LUN is aligned with the block size the storage subsystem uses. A sector size is 512 bytes. So the start sector of a partition should always be the multiple of the number of 512-byte blocks in a LUN block.

For the following example, the Oracle ZFS Storage ZFS volume is set up for 8k, so the start sector of the partition should be a multiple of 16.

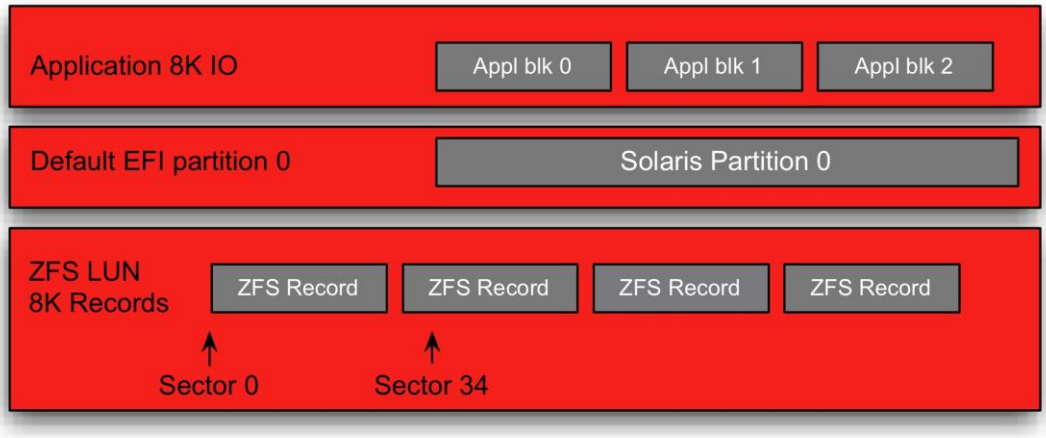


Figure 18. Oracle Solaris default EFI partition schema on LUN from an Oracle ZFS Storage Appliance

If the start sector created by a partitioning tool is 34, there is a misalignment. For 128k ZFS block sizes, the start sector should be multiples of 256.

Different OS types use different partitioning schemes. The x86-based operating systems use a partitioning scheme based upon the old DOS fdisk tool. Linux and VMware still use this type of partitioning scheme.

Industry best practice is moving towards recommending a 1 MB partition offset scheme alignment, so use a start sector of a partition that is a multiple of 2048. For a ZFS-based LUN/volume, select a partitioning offset (start sector) that is at least the same size as the largest ZFS block size, currently 128k.

The block size used by ZFS for a specific LUN is specified when creating the LUN in the following dialog window:

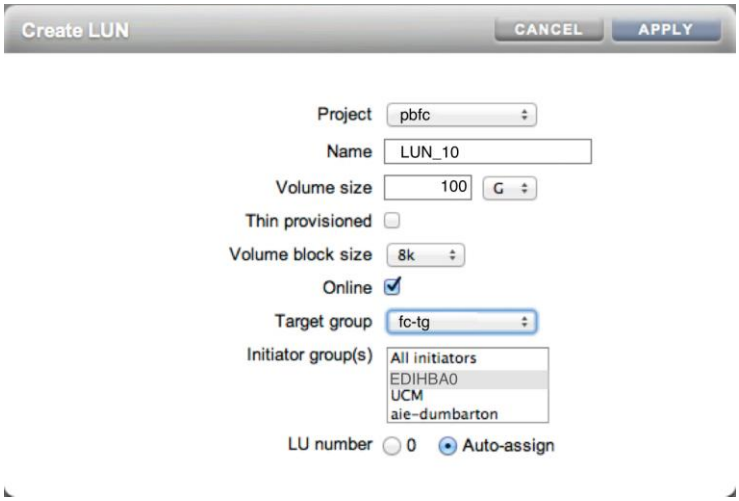


Figure 19. Specifying ZFS block size for a LUN

Oracle Solaris Partitioning

Oracle Solaris uses two partitioning schemes, the Storage Management Initiative (SMI) and Extensible Firmware Interface (EFI) standards. SMI originates from Oracle SPARC Solaris. It expresses the start and end of the partition in cylinders. The EFI standard uses a sector-based scheme.

The following code shows a partition creation designating an EFI label.

```
format>label
[0] SMI Label
[1] EFI Label
Specify Label type[1]: 1
Ready to label disk, continue? y

format> partition
       ~~~~~~~~~~~~~~
partition> print
Current partition table (original):
Total disk sectors available: 209698782 + 16384 (reserved sectors)

Part      Tag    Flag           First Sector         Size              Last Sector
  0        usr    wm           34            99.00GB          207618081
```

```
  1 unassigned    wm             0                0                     0
  2 unassigned    wm             0                0                     0
  3 unassigned    wm             0                0                     0
  4 unassigned    wm             0                0                     0
  5 unassigned    wm             0                0                     0
  6 unassigned    wm             0                0                     0
  7 unassigned    wm             0                0                     0
  8   reserved    wm     209698783             8.00MB          209715166

partition> 0
Part      Tag    Flag     First Sector          Size          Last Sector
  0        usr    wm                34       99.00GB           207618081
Enter partition id tag[usr]:
Enter partition permission flags[wm]:
Enter new starting Sector[34]: 256
Enter partition size[207618048b, 207618303e, 101376mb, 99gb, 0tb]:
partition> pri
Current partition table (unnamed):
Total disk sectors available: 209698782 + 16384 (reserved sectors)

Part      Tag    Flag     First Sector           Size                 Last Sector
  0        usr    wm          256           99.00GB          207618303
  1 unassigned    wm            0              0                     0
  2 unassigned    wm            0              0                     0
  3 unassigned    wm            0              0                     0
  4 unassigned    wm            0              0                     0
  5 unassigned    wm            0              0                     0
  6 unassigned    wm            0              0                     0
  7 unassigned    wm            0              0                     0
  8   reserved    wm    209698783             8.00MB          209715166
partition>
```

When using more than one partition, the EFI scheme provides a finer granularity to guarantee block alignment. The default start of the first EFI-based partition is 34. However, this causes misalignment for any ZFS block size. To assure alignment for any chosen ZFS block size, base the start sector number on ZFS maximum block size, 128K. This requires a start sector of 256 for the first partition.

Oracle Solaris on x86 platforms only supports the EFI-based partition schema. It must be used on top of the fdisk-based PC partitioning scheme. Note that fdisk carvings are called partitions here and the EFI schema are called slices. So there are one or a number of slices within an fdisk partitioning schema. Both schemas must be set up in such a way that each partition start and slice start within a partition is aligned with the ZFS block size. For further PC partitioning details, see the following sections on "Windows Partitioning," "VMware Partitioning," and "Linux Partitioning."

In an Oracle Solaris environment, you can create and modify the PC partitioning schema using the Gnome Partition Editor (GParted) tool, `(g)parted`.

The following illustration shows a graph for read I/Os coming in from FC and read I/Os from disk.

In the first period, an EFI-labeled LUN is used with a partition starting at sector 256. For the second period, the default EFI-generated partition start of sector 34 is used.

Notice in the first period there is a one-to-one relationship between the number of I/Os from the disk and the number of I/Os on the FC LUN (1850 IOPs).

In the second period, for each read on the FC LUN, two disk read I/Os are generated and, as a result, the number of I/Os on the FC LUN is less than when the FC I/O blocks are aligned with the ZFS block sizes.

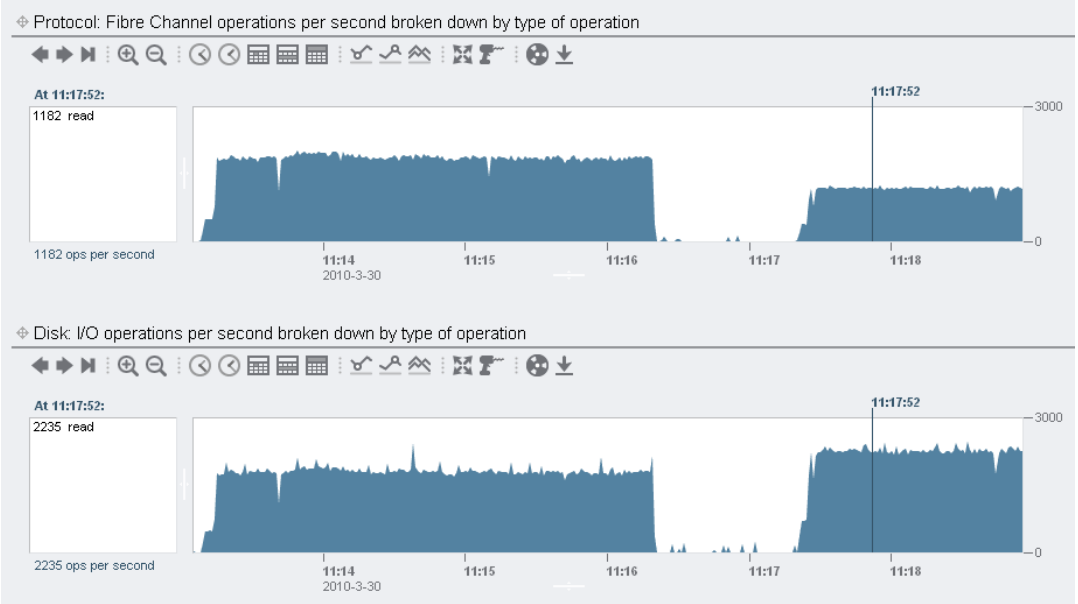


Figure 20. Comparison of 256 and 34 start sectors for EFI labeled LUN

Windows OS Partitioning

Partitioning alignment varies according to the installed version of the Windows operating system.

Windows Server 2008 and Windows Vista

In Windows Server 2008 as well as Windows Vista, partitioning alignment is by default set to 1 MB (for disks larger than 4 GB). You can check the value used in the registry under the following reference:

HKLM\SYSTEM\CurrentControlSet\Services\VDS\Alignment

Windows Server 2003 and Earlier

Partitions created in a Windows environment up to and including Windows 2003 Server are by definition not aligned. The default used start of the partition table is 32256 bytes. For Windows dynamic disks, use the command `dmdiag -v` to check the partition offset. For basis disks, use the `wmic` command.

Use the diskpart tool to create partitions on the created FC LUN. The following template can be used to set the partition offset, assign a drive letter, and specify a file allocation unit size (unit=<*xx*K>).

```
Diskpart
list disk
select disk <DiskNumber>
```

```
create partition primary align=<Offset_in_KB>
assign letter=<DriveLetter>
format fs=ntfs unit=64K label="<label>" nowait
```

Use a recommended offset value of at least 128K or multiples of that.

Note that the choice of the file allocation unit size depends on the application recommendation for how to use the Windows file system. Use a ZFS block size of the same value.

VMware Partitioning

In a VMware environment, the virtual machines use the Virtual Machine File System (VMFS) and are created within a PC fdisk partition. It is important that the created partitions are aligned with the underlying Oracle ZFS Storage Appliance's ZFS structure. When creating VMFS partitions using VMware's ESX hypervisor, defaults vary slightly and depend on the ESX version.

Data file systems for a guest operating system are stored in Virtual Machine DisK format (VMDK) on top of a VMFS with which the file systems must be aligned. Guest operating systems can also use raw device mappings. The alignment of partitions on raw device mappings is the same as for an OS directly using a LUN. Alignment of the guest OS boot partitions is not required.

To simplify the alignment model, use a 256-sector offset for both the partition on the LUN that holds the VMFS and the partitions on top of the VMFS that hold the VMDKs of the guest OS.

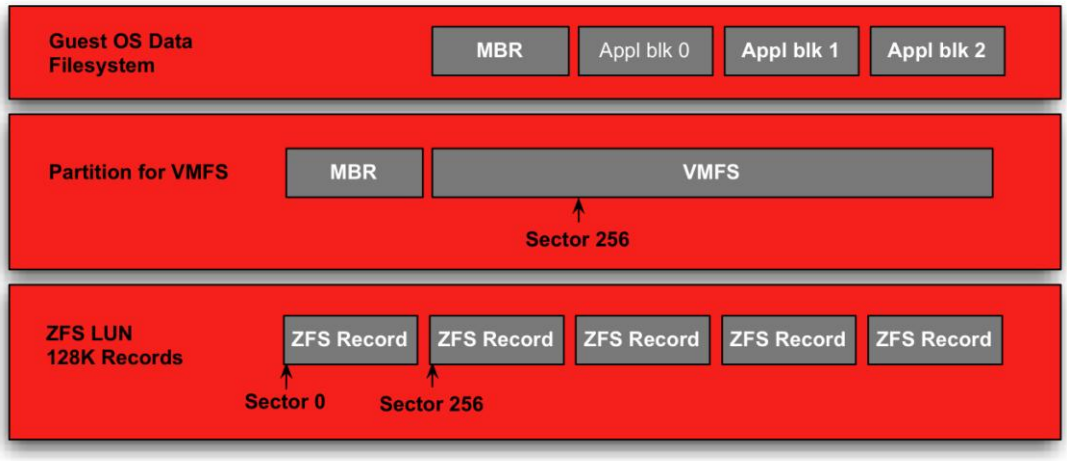


Figure 21. VMware VMFS and VMDK partition alignment

VMware ESX 3.0

Use the fdisk tool in expert mode to specify the required start block of partition 1.

VMware ESX 4.0 and ESXi 4.0

The vmkfstools program in the vSphere Client automatically aligns the partitions along the 64KB boundary. This will not cover alignment for all possible ZFS block sizes. The use of a 128K boundary is recommended.

VMware VMFS Block Sizes

The size of the VMFS block size determines the maximum size of a VMFS instance; there is no relationship between the I/O sizes used by the guest OS (file system) and the VMFS block size. Guest OS read and write operations are transparently passed through to the external storage subsystem by the vmkernel. So the value of the used ZFS block size should be determined by the I/O size used by the guest OS file system/application. Many file systems use a 4K block size.

Linux Partitioning

The Linux fdisk partitioning tool also uses cylinders to define the start and end of a partition. The cylinder start position and size should align with the blocks of the underlying ZFS file system. The Linux fdisk tool's `-S` option allows you to specify the number of sectors per track. Using a value that is a multiple of the ZFS block size assures alignment.

However, fdisk's `-S` option does not accept values beyond 63. In most cases, 63 is the value used by default. This will always cause misalignment for ZFS block sizes greater than 1 KB.

A practical workaround is to specify 32 and use the command `fdisk -u` to specify the start value of a partition expressed in sectors. Specify 256 for the first partition and a multiple of 256 for all the following partitions using the `+`*n*, where *n* is the number of sectors in the partition. This workaround will cover all ZFS block sizes.

Note: The fdisk routine might display warnings that 'partition does not end (or start) on cylinder boundary.' You can safely ignore those warnings, as cylinders are just an artificial construct of the tool.

```
root@petertest /root % fdisk -u -S 32 /dev/sda

The number of cylinders for this disk is set to 5140

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First sector (32-41942399, default 32):256
Using default value 256
Last sector, +sectors or +size{K,M,G} (256-41942399, default 41942399):
Using default value   419424399

Command (m for help): p

Disk /dev/sda: 21.4 GB, 21474836480 bytes
255 heads, 32 sectors/track, 5140 cylinders, total  41942400 sectors
Units = sectors of1* 512 bytes
```

```
Disk identifier: 0x000da626

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *         256      41942399    106080   83  Linux

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

Calling ioctl() to re-read partition table.
Syncing disks.
root@sysresccd /root % fdisk -l -u /dev/sda

Disk /dev/sda: 21.4 GB, 21474836480 bytes
255 heads, 32 sectors/track, 5140 cylinders, total  41942400 sectors
Units = sectors of 1 * 512 = 512 bytes
Disk identifier: 0x000da626

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *         256     41942399    2097120   83  Linux
root@petertest /root %
```

As previously mentioned, industry practice is moving to a 1 MB size partition offset scheme alignment. So use a start sector of a partition that is a multiple of 2048.

SSD Cache Type Guidelines

Reference Test Setup

Performance tests provide a better understanding of the behavior of various cache configurations. The I/O workloads vary, depending on the type of application and the type of workload the application is executing. An application might show a high random and write percentage workload during the day and sequential, read-oriented workload during the night while performing backups on some form of data mining activity.

To present a range of OLTP type workloads, each corner of the workload spectrum is used to measure I/O behavior, as shown in the following table. A complete test set requires execution of eight different workloads to obtain I/Os per second results.

TYPE OF I/O	LOCALITY OF DATA	SEQUENTIAL	RANDOM
100% Read	100% Cached	IOPs	IOPs
	Non-cached	IOPs	IOPs
100% Write	100% Cached	IOPs	IOPs
	Non-cached	IOPs	IOPs

For locality of data categories, data caching in the adaptive replacement cache (ARC) of the Oracle ZFS Storage Appliance was measured, with either 100% or extremely low hits. To ensure a 100% cache hit, I/O was executed to/from a small part of the LUN; in this case, 1 GB. A non-cache hit situation was created to execute I/O over the whole area of the LUN. The size of the LUN was 100 GB, which is much bigger than the cache size used in the Oracle ZFS Storage Appliance.

To measure the difference between asynchronous and synchronous writes, the write test was run on the LUN with and without the dsync open flag option.

Vdbench was used as performance test tool. This tool is extremely flexible and allows for a well specified and controlled workload. The workload that was used is shown in the VDBENCH parameter file in Appendix B.

To ensure that the workload would not cause a queue overrun for the LUN under test, a maximum of 32 threads was used. For an added comparison point, an OLTP type workload was executed, using a load with 75% and 25% reads.

The tests were run on a LUN from a pool with the following configurations:

- Without any SSD cache devices

- With read optimized cache devices (L2ARC)
- With write optimized cache devices (ZIL), and
- With combined read and write optimized cache devices.

Note: The test performance setup does not indicate the maximum performance of a Oracle ZFS Storage Appliance used in an FC target environment. It merely gives a set of figures to compare Oracle ZFS Storage Appliance ARC, L2ARC, and ZIL behavior for extreme workload corner cases.

One extra parameter was taken into account in the tests: turning the Oracle ZFS Storage Appliance write caching on and off for the LUN under test. (See Write Cache Considerations in the first section, Using the Oracle ZFS Storage Appliance for FC LUNS.) This option, as illustrated in figure 13 of section Write Cache Considerations, turns the use of the ARC on or off for write caching.

Note: Using the LUN write caching option is ***strongly*** discouraged. Using this option may cause data integrity issues. Users should understand this risk when enabling this option and ensure that data can be recovered from an alternative source.

The test LUN was 100 GB, with 8 KB ZFS block size from a pool using a full Sun Storage J4400 rack (24 devices) in a mirrored RAID configuration. One read-optimized cache device and two (mirrored) write optimized cache devices were used for ARC2 testing.

Interpreting Test Results

The following section explores the performance test results for the various configurations, in order to draw conclusions about optimal settings for different performance objectives.

Oracle ZFS Storage Appliance Write Cache Usage

The following graph compares the results of tests with a LUN with no SSD type cache devices that varied the LUN write cache option value.

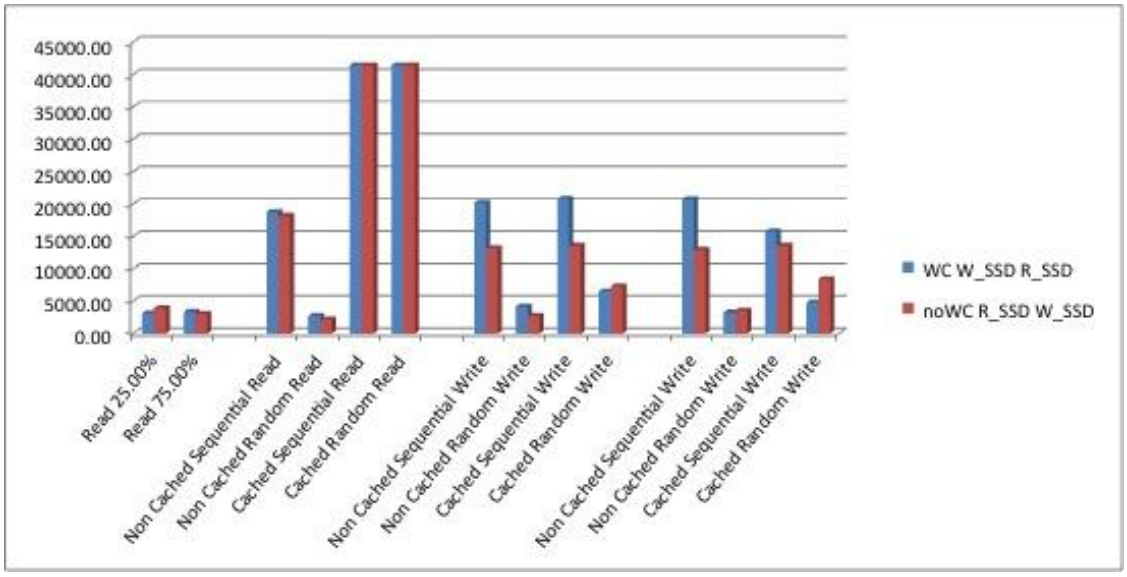


Figure 22. Comparing LUN write cache options with LUN, no SSD devices

As expected, read performance is not influenced by enabling or disabling the LUN write cache setting. For write operations, there is a significant difference.

The following graph shows the comparative performance when a write-optimized SSD device is added to the pool in which the LUN is configured.

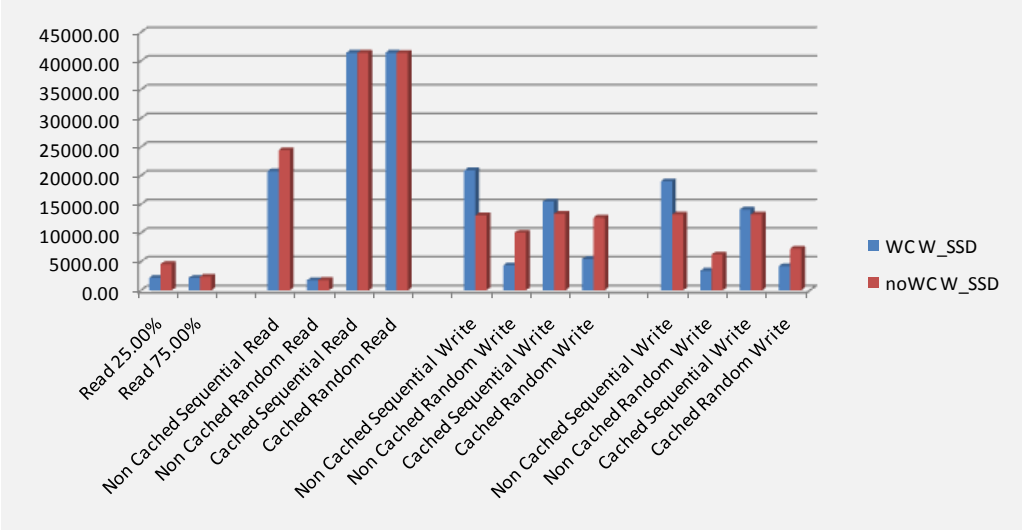


Figure 23. Comparing LUN write cache options for a LUN with added SSD device

Using the Write Optimized SSD option compensates significantly for the exclusion of the ARC for write caching. This means that when write performance and data integrity are important, using the Write Optimized SSD option provides the same performance levels as setups in which the LUN write caching is switched on. More importantly, using the Write optimized SSD option *ensures data integrity* of

the system with the performance levels of a system using local memory cache speeds. The number of write cache devices must be sized to the expected 'locality' of the application workload.

SSD Cache Device Type Comparison

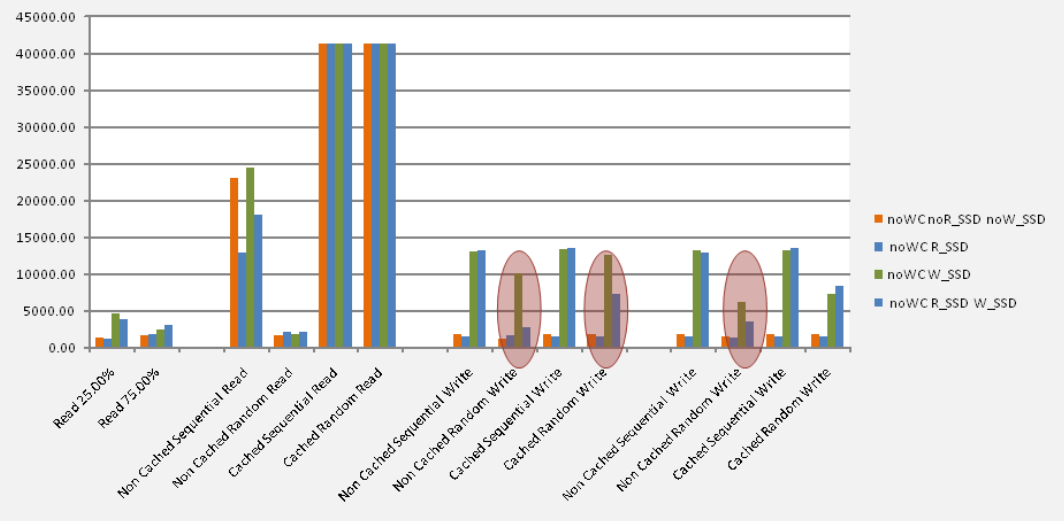


Figure 24. LUN performance using different SSD cache type options

Comparing the results of tests run with various SSD cache types yields better insight on when to use which type of SSD cache type devices.

An obvious observation is that for cached read I/O type workloads, SSD type cache devices do not seem to matter. Note, however, that this is true for a load on a single LUN, as used in this test. All read data fits in the Oracle ZFS Storage ARC. When multiple LUNs (and/or volumes) are used, read type SSDs can be used to create a second-level read cache (L2ARC).

Another very important conclusion is that when dealing with high write random-oriented type workloads that have a high cache hit rate in the ARC, it is better to set 'cache device usage' to 'meta data only' or 'none.

OLTP Type Workload

The following graph shows tested workloads of 75% read in the foreground (first row), and 25% read in the background (second row). The graph further illustrates the previous conclusions and observations for the tested read function.

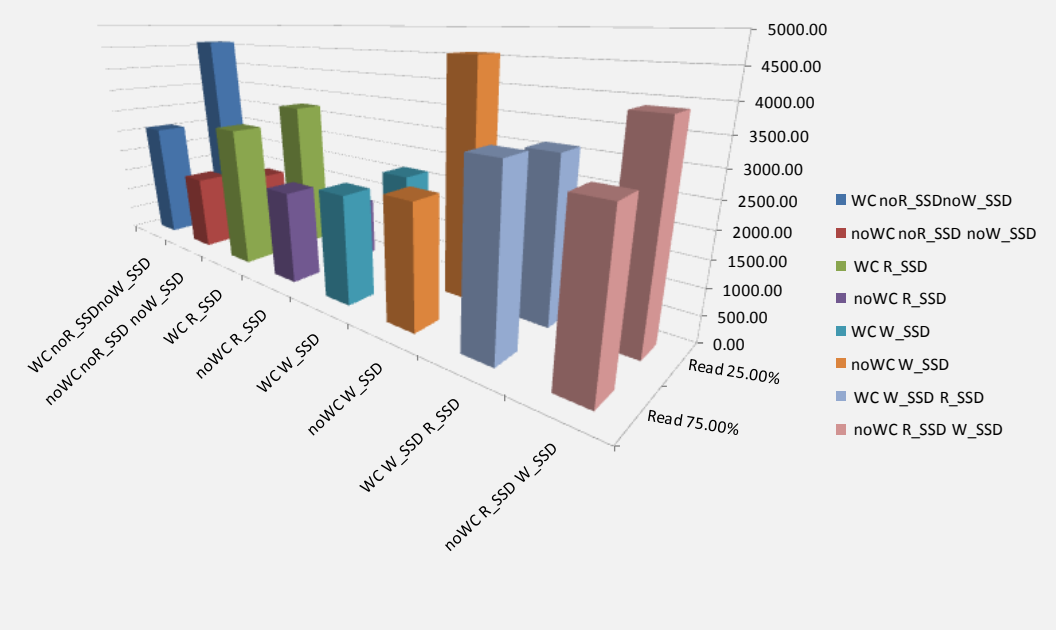


Figure 25. Comparing LUN performance for SSD type cache options for OLTP type workloads

Conclusions

The results show two major factors to take into account when configuring SSD type cache devices;

- For random write-intensive I/O load, set 'cache device usage' to 'meta data' or 'none.' (See figure 27.)
- . If you are not using the LUN write cache option, you can compensate for the performance impact by configuring the right type and number of SSD type cache devices.

In general, tune pools to the I/O performance characteristics of the LUNs/volumes to be used. Follow the recommendations in the BUI of the Oracle ZFS Storage Appliance when selecting a RAID profile option for a pool you are creating. The BUI provides a short description for each of the available profile options.

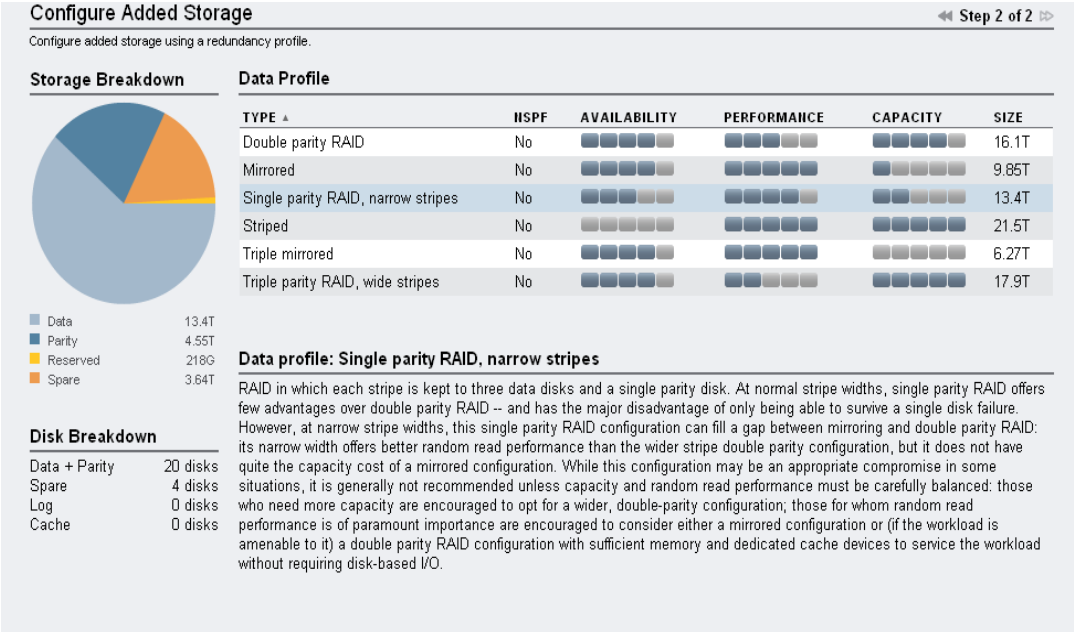


Figure 26. BUI, data profile options and description

It is possible to influence the use of SSD type cache devices on a per LUN/volume basis. As part of the properties options for either a share or a single LUN/volume, the behavior of read and write for SSD cache devices can be specified as listed in the following table:

PROPERTY	OPTION	RESULT
Cache device usage	All data and metadata	Read type SSD used for all data, this project/LUN/volume
	Metadata only	Read type SSD used for all metadata, this project/LUN/volume
	Do not use cache devices	Read type SSD not used for this project/LUN/volume
Synchronous write bias	Latency	Write type SSD enabled for this project/LUN/volume
	Throughput	Write type SSD not used for this project/LUN/volume

The SSD-related properties can be specified in the Properties window for either the project or LUN/volume.

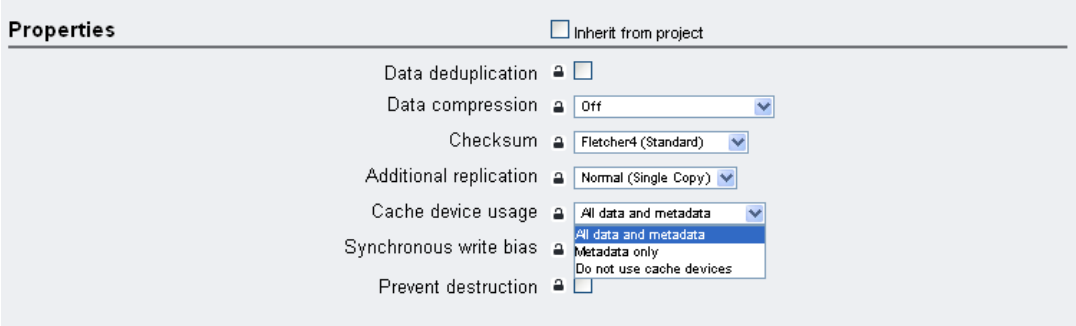


Figure 27. Specifying SSD cache type behavior in share/volume/LUN properties

Appendix A: The vdbench Parameter File

The vdbench version 5.02 tool was used to generate the performance figures used in this paper. One thing to be aware of is that ZFS returns read requests directly to the client without reading from disk or cache if the block requested was never written. This will obviously give very positive read figures but not very realistic ones. For each LUN created on the Oracle ZFS Storage Appliance a dd write stream was executed on that LUN before starting any performance tests on that LUN.

The following vdbench workload script was used:

```
# LUNs
sd=LUN1_8k,hitarea=1GB,threads=32,lun=/dev/rdsk/c16t600144F08C9A347B00004BACF5290008d0s
2
sd=LUN1_8k_sync,hitarea=1GB,threads=32,openflags=o_dsync,lun=/dev/rdsk/c16t600144F08C9A
347B00004BACF5290008d0s2



# Following measurements:
#
#          cached                non-cached
# read     sequential, random    Sequential, Random
# write    sequential, random    Sequential, Random
#
# Cached is done by using 1GB of the LUN for the measurement, as defined by hitarea in
sd definition, see above
#
# Only 8kB blocks are used.
# To simulate a DB IO, last test is run for 75% and 25% reads.
#  with O_DSYNC using 8kB blocks

wd=smallio,sd=(LUN1_8k)
wd=smallio_sync,sd=(LUN1_8k_sync)


#
rd=ReadsSmallIO,wd=smallio,warmup=200,rdpct=100,forrhpct=(0,100),forseekpct=(0,100),for
xfersize=(8k),forthreads=(32),iorate=max,elaps=90,interval=10,pause=10

rd=WritesSmallIO,wd=smallio,warmup=200,rdpct=0,forwhpct=(0,100),forseekpct=(0,100),forx
fersize=(8k),forthreads=(32),iorate=max,elaps=90,interval=10,pause=10
rd=WritesSmallIO_sync,wd=smallio_sync,warmup=200,rdpct=0,forwhpct=(0,100),forseekpct=(0
,100),forxfersize=(8k),forthreads=(32),iorate=max,elaps=90,interval=10,pause=10


# DB mix load
rd=DBLoad,wd=smallio_sync,warmup=200,rdpct=75,forrhpct=(25,75),whpct=5,seekpct=100,xfer
size=8k,forthreads=(32),iorate=max,elaps=90,interval=10,pause=10
```

Appendix B: VMware ALUA Support Setup

The drivers in VMware ESX 4.0 do not recognize the Oracle ZFS Storage Appliance FC multipath ALUA capabilities by default. The drivers in ESX 4.1 are ALUA aware.

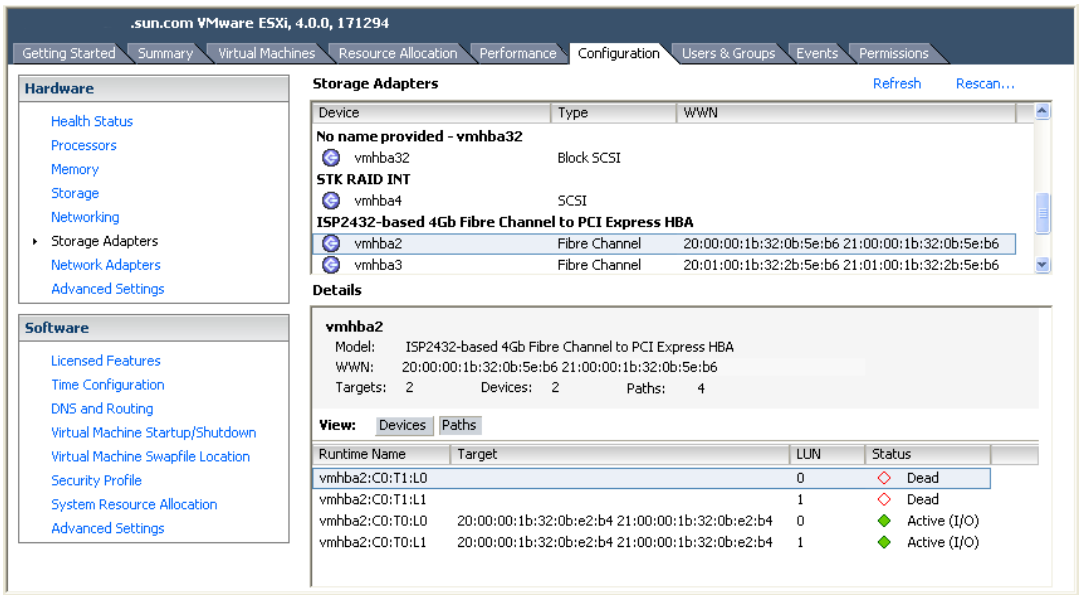


Figure 28. FC LUN path in VMware

When an FC LUN is made available, the standby path shows 'Dead' as status. In order to get VMware to recognize the Oracle ZFS Storage Appliance FC LUN standby path, a configuration rule must be added to the SATP plugin. This rule must be added to each VMware ESX(i) server that will be connected using Fibre Channel to a Oracle ZFS Storage Appliance cluster configuration.

The following described commands are executed from the ESX(i) server's command-line interface. Either use SSH to gain access to the server or use the CLI interface on the console.

Verifying the Current SATP Plugin Rule

Execute the following `esxcli` command:

```
~ # esxcli nmp device list
naa.600144f08c9a347b00004be43587001f
    Device Display Name: SUN Fibre Channel Disk (naa.600144f08c9a347b00004be43587001f)
    Storage Array Type: VMW_SATP_DEFAULT_AA
    Storage Array Type Device Config:
    Path Selection Policy: VMW_PSP_FIXED
    Path Selection Policy Device Config:
{preferred=vmhba2:C0:T1:L0;current=vmhba2:C0:T0:L0}
    Working Paths: vmhba2:C0:T0:L0
```

Find the `SUN Fibre Channel Disk` entries in the command output. The `Storage Array Type` is shown as `VMW_SATP_DEFAULT_AA`. This array type definition does not have ALUA capabilities.

Determining Vendor and Model `ident` Information

The vendor and model `ident` information can be found in the messages file of the VMware kernel.

```
~ # fgrep SUN /var/adm/messages | fgrep Storage
May  7 18:00:06 vmkernel: 1:00:51:19.169 cpu13:5070)ScsiScan: 839: Path
'vmhba2:C0:T0:L0': Vendor: 'SUN     '  Model: 'Sun Storage 7410'  Rev: '1.0 '
```

Using the previous command returns a number of lines of text for the Oracle ZFS Storage Appliance FC LUNs detected by the VMware kernel. Note that for each model of a Oracle ZFS Storage Appliance, the model number information will be different.

Adding and Verifying the SATP Configuration Rule

The text following vendor and model has to be used to make VMware recognize the Oracle ZFS Storage Appliance ALUA capabilities by executing the following command:

```
~ # esxcli nmp satp addrule -s VMW_SATP_ALUA -e "Sun Storage 7410" -V "SUN" -M "Sun
Storage 7410" -c "tpgs_on"
~#
```

In this example the Oracle ZFS Storage Appliance 7410 model has been used. Verify if the rule was correctly added by executing the following command:

```
~ # esxcli nmp satp listrules | fgrep SUN | fgrep ALUA
VMW_SATP_ALUA       SUN      Sun Storage 7410      tpgs_on        Sun Storage 7410
~#
```

Verifying the FC LUN ALUA Path Status on the Oracle ZFS Storage Appliance

Reboot the ESX server, or if there are no FC LUNs discovered yet, proceed with re-scanning the FC HBAs to add new LUNs.

After server boots, recheck the plugin that is in use. Once enabled all newly discovered LUNs will be captured by the ALUA plugin.

```
~ # esxcli nmp device list
naa.600144f08c9a347b00004be43587001f
    Device Display Name: SUN Fibre Channel Disk (naa.600144f08c9a347b00004be43587001f)
    Storage Array Type: VMW_SATP_ALUA
    Storage Array Type Device Config:
{implicit_support=on;explicit_support=off;explicit_allow=on;alua_followover=on;{TPG_id=
0,TPG_state=AO}{TPG_id=1,TPG_state=STBY}}
    Path Selection Policy: VMW_PSP_MRU
```

```
    Path Selection Policy Device Config: Current Path=vmhba3:C0:T0:L0
    Working Paths: vmhba3:C0:T0:L0
```

Check the path status. In this configuration example there are four paths for one LUN: two active, two standby. The same configuration setup was used as shown in the section SAN Configuration and Zoning Setup.

```
~ # esxcli nmp path list

fc.2001001b322b5eb6:2101001b322b5eb6-fc.2001001b32335c63:2101001b32335c63-
naa.600144f08c9a347b00004be43587001f
    Runtime Name: vmhba3:C0:T1:L0
    Device: naa.600144f08c9a347b00004be43587001f
    Device Display Name: SUN Fibre Channel Disk (naa.600144f08c9a347b00004be43587001f)
    Group State: standby
    Storage Array Type Path Config: {TPG_id=1,TPG_state=STBY,RTP_id=256,RTP_health=UP}
    Path Selection Policy Path Config: {non-current path}

fc.2001001b322b5eb6:2101001b322b5eb6-fc.2001001b322be2b4:2101001b322be2b4-
naa.600144f08c9a347b00004be43587001f
    Runtime Name: vmhba3:C0:T0:L0
    Device: naa.600144f08c9a347b00004be43587001f
    Device Display Name: SUN Fibre Channel Disk (naa.600144f08c9a347b00004be43587001f)
    Group State: active
    Storage Array Type Path Config: {TPG_id=0,TPG_state=AO,RTP_id=2,RTP_health=UP}
    Path Selection Policy Path Config: {current path}

fc.2000001b320b5eb6:2100001b320b5eb6-fc.2000001b32135c63:2100001b32135c63-
naa.600144f08c9a347b00004be43587001f
    Runtime Name: vmhba2:C0:T1:L0
    Device: naa.600144f08c9a347b00004be43587001f
    Device Display Name: SUN Fibre Channel Disk (naa.600144f08c9a347b00004be43587001f)
    Group State: standby
    Storage Array Type Path Config: {TPG_id=1,TPG_state=STBY,RTP_id=257,RTP_health=UP}
    Path Selection Policy Path Config: {non-current path}

fc.2000001b320b5eb6:2100001b320b5eb6-fc.2000001b320be2b4:2100001b320be2b4-
naa.600144f08c9a347b00004be43587001f
    Runtime Name: vmhba2:C0:T0:L0
    Device: naa.600144f08c9a347b00004be43587001f
    Device Display Name: SUN Fibre Channel Disk (naa.600144f08c9a347b00004be43587001f)
    Group State: active
    Storage Array Type Path Config: {TPG_id=0,TPG_state=AO,RTP_id=1,RTP_health=UP}
    Path Selection Policy Path Config: {non-current path}
~#
```

Restart the vSphere Client. The path information can be found under the 'Configuration' tab, using the devices view, selecting a Fibre Channel disk and using the 'Manage Path' option.

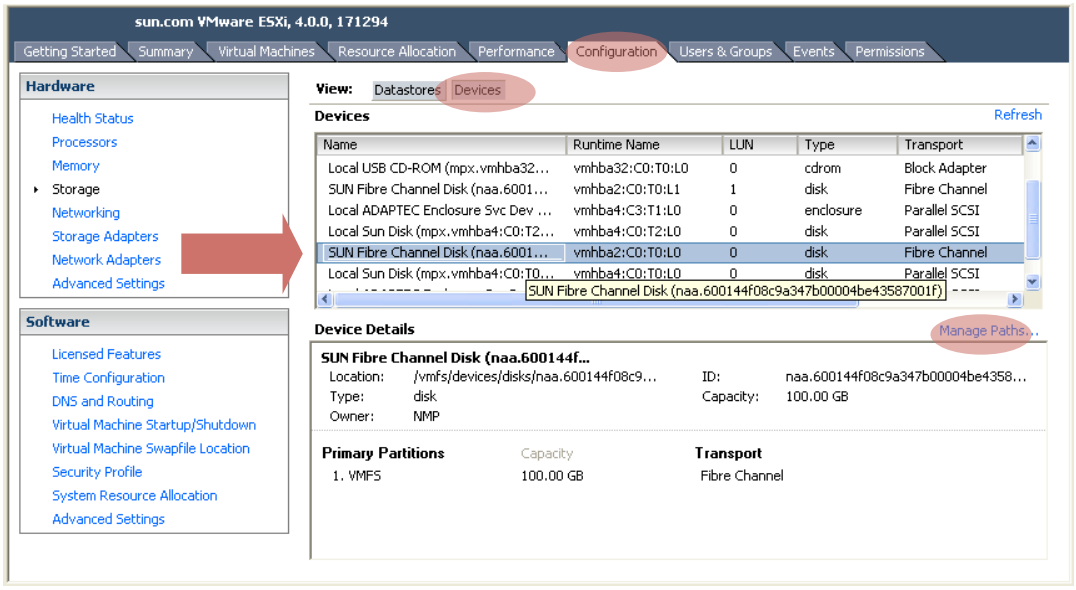


Figure 29. VMware FC LUN device information

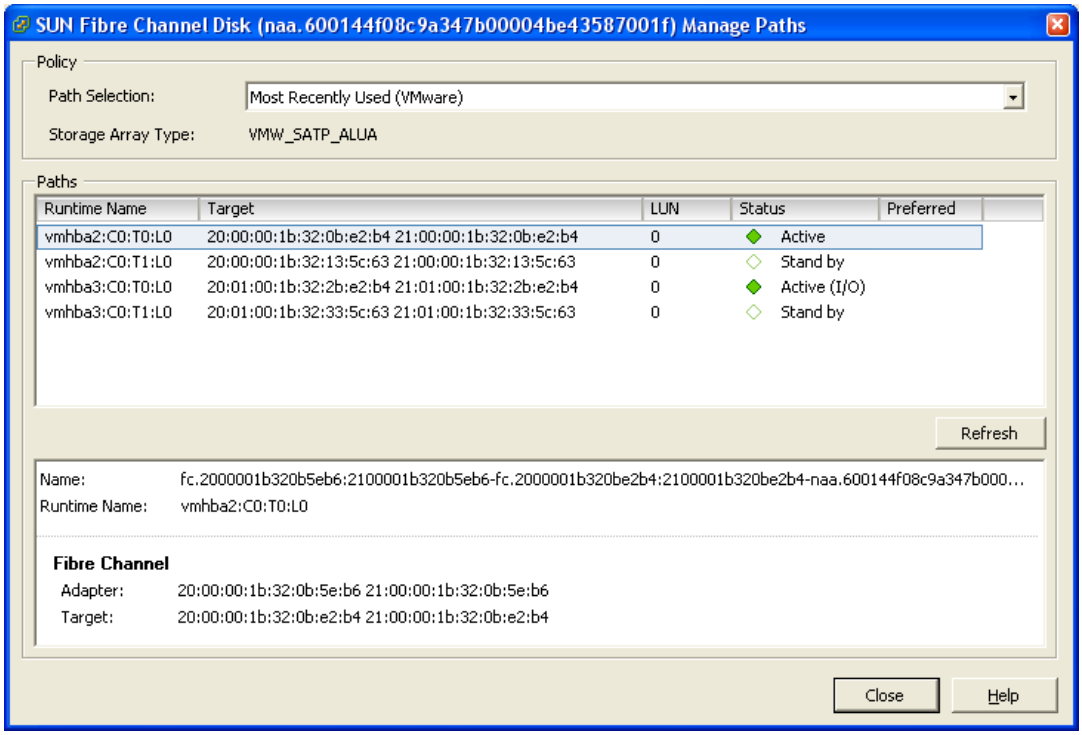


Figure 30. VMware FC LUN path information

Appendix C: References

Reference Documentation

Oracle ZFS Storage Appliance Documentation

Blogs

Oracle Blogs portal

Aligning Flash Modules for Optimal Performance

A Quick Guide to the ZFS Intent Log (ZIL)

Brendan Gregg, ZFS L2ARC

White Papers

"Aligning Partitions to Maximize Storage Performance" and other Sun NAS Storage white papers
http://www.oracle.com/technetwork/server-storage/sun-unified-storage/documentation/index.html

Understanding the Use of Fibre Channel in the Oracle ZFS Storage Appliance
4, v 1.1
Author: Peter Brouwer

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200

oracle.com

Oracle is committed to developing practices and products that help protect the environment

Hardware and Software, Engineered to Work Together