

# APPLICATION NOTE

## Xyratex 54xx RAID Systems and Quantum's StorNext File System

v1.2  
19 December 2007

---

Contributors: Dick Dirisio (DASDI)  
Brian Morsch (Quantum)  
Document Owner: Bill Krempp (Xyratex)

Xyratex  
1804 Centre Point Circle  
Suite 112  
Naperville, IL 60563 USA

[www.Xyratex.com](http://www.Xyratex.com)

**Copyright © Xyratex Technology Limited. 2007**

### COPYRIGHT AND INTELLECTUAL PROPERTY NOTICE

The content of this Application Note is protected by the copyright laws of the United States of America and the international copyright laws and agreements. Xyratex Technology Limited, its subsidiaries and affiliates (collectively, "Xyratex") own (or have permission to use) all intellectual property rights in relation to this Application Note and its content (including, but not limited to, all trademarks and copyright). Without the prior written authorization of Xyratex, no part of this Application Note may be used, disclosed, reproduced, displayed, copied, translated, adapted, broadcast, or republished in any form including (without limitation) distribution or storage in a system for retrieval. Any information included in this Application Note is disclosed "As Is" and Xyratex makes no warranty of any kind with respect to the accuracy of such disclosed information or its suitability for any particular use.

Xyratex Proprietary and Confidential

**Table of Contents**

1.	Introduction .....	3
1.1.	Architecture.....	4
1.2.	Test Configuration .....	5
1.2.1.	Physical Test Setup .....	5
1.2.2.	RAID Storage Systems .....	5
2.	StorNext Configuration and Testing .....	7
2.1.	FC Path Failure and Recovery .....	7
2.2.	RAID Controller Failure and Recovery .....	8
2.3.	Metadata Controller (MDC) Failure and Recovery .....	9
3.	Array Configuration and Optimization .....	13
3.1.	F5412E Array Configuration .....	13
3.1.1.	Arrays and LUNs .....	13
3.1.2.	SNFS Sequential Read/Write Performance .....	13
3.2.	F5404E Array Configuration .....	14
3.1.2.	Arrays and LUNs .....	14
3.1.3.	SNFS Sequential Read/Write Performance .....	14
3.3.	Array Optimization .....	15
3.4.	Chunk Optimization.....	15
3.5.	Write Cache Optimization .....	16
3.6.	Read Cache Optimization .....	16
4.	Performance Counters.....	17
4.1.	Workload.....	17
4.2.	Cache.....	17
4.3.	Array .....	18
5.	Conclusions .....	19
6.	Appendix.....	20
a.	StorNext SNFS Control Files.....	20
i.	snfs2.cfg .....	20
ii.	cvlabels .....	22
iii.	cvpaths .....	23
iv.	fsmlist.....	23
v.	fsnameservers.....	23
vi.	dpservers .....	24

## 1. Introduction

The Xyratex F54xxE systems are based on 4Gb Fibre channel. It provides the bandwidth needed for high data throughput, video editing to database intensive, online transaction processing. Internally, the Xyratex F54xxE systems leverage SAS full duplex, point-to-point architectures with dedicated connections for maximum data throughput.

As a high performance alternative to NAS, Quantum's StorNext® file system software enables high speed, heterogeneous data sharing so that workflow operations run faster and more reliably. When combined with the StorNext Storage Manager, data sharing, management and retention are combined in a single solution.

This application note describes the RAID configurations and tests used to qualify Xyratex's F54xxE RAID systems with Quantum's StorNext file system (SNFS). It is intended for System Administrators planning StorNext SNFS installations using the F54xxE RAID controllers and as a "quick start" guide to configuring and designing additional test procedures for larger RAID configurations.

Section 1 A brief overview of the controllers' architecture, the RAID system configurations and the physical connections between the RAID controllers and the servers.

Section 2 The StorNext pathing configuration and a detailed description of the failure modes tested. A full listing of the StorNext configuration files is included in the appendix.

Section 3 The physical and logical configuration of the arrays used in the tests, sequential R/W performance of each config and overview of the RAID controllers cache subsystem.

Section 4 Statistics available on the F540xxE RAID controllers. Sys Admins can use this data to improve overall system performance.

Section 5 A summary of the testing and links to additional information.

Section 6 Appendix – StorNext configuration files.

## 1.1. Architecture

### RAID Controllers

With both controllers installed and operational, the F54xxE controllers operate in a truly active/active manner. Any host IO ingressed by a RAID controller is processed at that RAID controller. Intelligent cache synchronisation techniques are used between both controllers to enable integral accesses of the logical drives in a mode where concurrent IO can be serviced by either controller for the same logical drive. Access to SCSI LUNs is completely symmetric and no preference is offered to any LUNs presented by the controller.

Each controller has two 4Gb/sec FC ports which can run at 4Gb/s, 2Gb/s or 1Gb/s. Each port is capable of operating in fabric, loop or point-to-point interconnects. A controller can present up to 512 LUNs for any given host. A full range of SAN mapping capabilities are provided by the controller for the user.

The F54xxE controllers take full advantage of the Serial Attached SCSI (SAS) interconnect. SAS utilises a multi lane configuration forming a tree of nodes in the controller's domain. Each lane is capable of operating at up to 3Gb/sec, thus providing up to 12Gb/sec of bandwidth for both drive and cache coherency traffic. Drive connections use 2 SAS lanes and cache coherency the remaining 2. Either RAID controller therefore has 6Gb/sec of bandwidth to the drives it operates.

SAS has the capability of supporting SATA drives via a tunnelling protocol known as STP. This allows for greater choices of drive types for the storage systems. The F54xxE can be configured with high performance SAS 15K and 10K drives, as well as 7.2K SATA drives (the F5404E only supports SATA disks). SAS offers native multi-ported drives for high availability configurations. SATA drives are inherently single ported but within the enclosure they are fitted with an interface board to allow either controller to access the drive,

### StorNext File System

The StorNext File System (SNFS) provides extremely high performance for a wide range of applications. This file system software includes multi-path detection and failover, load balancing capabilities, fine grained control over data placement and metadata controller (MDC) redundancy. The 3.x release of StorNext includes a number of new features to simplify deployments and exponentially expand the number of nodes which can be included in a cluster.

- Single network support. This allows data and metadata traffic on the same network thus reducing by 50% the cabling and switch ports required for deployment.
- LAN client support. Eliminates the requirement for all StorNext clients to have a FC connection to the storage SAN. This can reduce infrastructure costs by over 90% when deploying StorNext in an HPC environment. With the LAN client installed, an HPC cluster compute node bypasses the NFS stack with improved performance and reliability at a fraction of the cost required for SAN clients.
- IPMI supported STONITH on common network. Allows StorNext to power off a "failed" node without the need for dedicated out-of-band hardware.

We will refer to various StorNext control files in this document which need to be edited and/or created. Most of these files can be found in the /usr/cvfs/examples and /usr/cvfs/config directories created during the StorNext installation. (This report assumes that you are familiar with the SNFS software. Please refer to the Quantum website and StorNext documentation if required.)

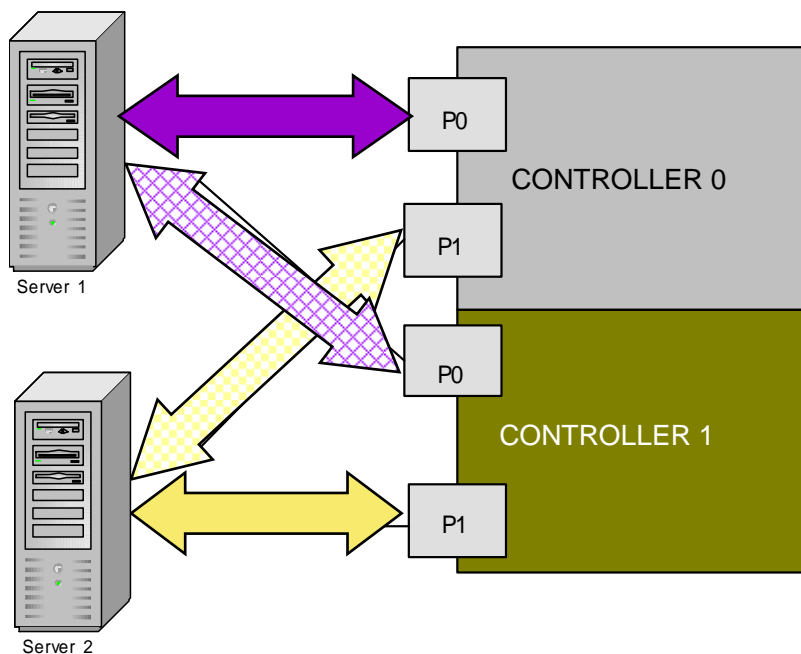
**1.2. Test Configuration**

The test configuration takes full advantage of the F54xxE controllers' ability to present LUNs to any hosts on any port while maintaining full cache coherency. Each host has a direct connect to one port on each of the RAID controllers. With all LUNs visible on each RAID port, this simple (and default) configuration meets the minimum requirements for StorNext's built-in multipath detection. System administrators should be aware of this default LUN visibility setting before attaching and powering up any servers attached to the RAID arrays.

**1.2.1. Physical Test Setup**

The basic test setup used for both RAID controllers is a 2-node StorNext cluster. Each cluster node is a StorNext SAN client and LAN gateway. One cluster node is designated as the active MDC server, the other cluster node is the standby MDC (refer to the StorNext documentation and Section 6 for details on configuring StorNext).

Each cluster node has a dual channel FC card installed. This FC card is direct attached to one port on each of the RAID controllers as shown in the diagram below.



Servers are running Novell's SLES10SP1 Linux distribution. They include 2 \* 2.8GHz Intel dual Xeon CPUs with an 800mhz FSB. HBA's from Q-logic (3xQLA2342, 1xQAL2462) provide dual fibre channel HBA connections. Driver version 8.01.07.5 was used for this testing. Default settings were used for all configurable qla2xxx driver parameters. Servers have 4GB of DDR333 memory. Quantum's StorNext 3.1 client/server software is installed on each system.

**1.2.2. RAID Storage Systems**

The F54xxE arrays used in testing were configured with 1GB cache/controller. The RAID controllers are running firmware version 3.3.0027 (F5404E) or 3.3.0029 (F5412E). Default cache settings are used on the arrays;

- **F5412E Array**
  - 18x750GB SATA disks, 7.2K RPM
  - 12x250GB SATA disks, 7.2 RPM
  - 3x73GB SAS disks, 15K RPM
  - Read-ahead ON (Auto per array)
  - Write-back ON (16MB per array)
  - Mirror-cache ON
  
- **F5448E Array**
  - 48x750GB SATA disks, 7.2K RPM
  - Read-ahead ON (Auto per array)
  - Write-back ON (16MB per array)
  - Mirror-cache ON

A detailed description of the RAID array configurations is included in Section 3.

## 2. StorNext Configuration and Testing

The StorNext file system provides robust fault detection, isolation and recovery. When properly configured, it can load balance IO across multiple paths and LUNs. StorNext configuration files control how the software uses multipath capable LUNs. In our direct connect test configuration, each RAID LUN has two paths (one per HBA port). StorNext control files (i.e. cvpaths) are edited so that one path is defined as ACTIVE and the 2<sup>nd</sup> path is PASSIVE. When a path fails, StorNext detects the failure, quiesces the ACTIVE path, checks status of the PASSIVE path, activates the PASSIVE path and reissues any failed block IO.

In our test configuration, ~half of the RAID LUNs (the even numbered LUNs) have their ACTIVE path mapped to the C0 controller and half (the odd numbered LUNs) have their ACTIVE path mapped to the C1 controller. This even-to-C0, odd-to-C1 configuration is simple to configure when using a 1:1 relationship between F54xxE raid arrays and LUNs. (If you need to define more than one LUN per array, you can use StorView to manage the external LUN ID's.)

By default, StorNext will find and activate all paths to a LUN. To over ride this behaviour, we must create a cvpaths file in /usr/cvfs/config to define the paths to each LUN. This file also defines which path is preferred. The cvpaths file used in test cluster snfs2 is included in Section 7.

### 2.1. FC Path Failure and Recovery

iozone is used to generate load to the local snfs file system from each cluster node.

```
demo3:/mnt/snfs2/demo3 # ./iozone -i0 -i1 -i2 -s1g -b64k -T -t16
```

On this node, all of the odd LUNs have their ACTIVE path mapped through the c1p0 port. The fibre channel cable attached to RAID port c1p0 is pulled. After 30s, the scsi driver starts reporting IO errors.

```
Dec 13 15:23:12 demo3 kernel: qla2400 0000:07:02.1: LOOP DOWN
detected (2).
Dec 13 15:23:43 demo3 kernel: sd 3:0:0:5: SCSI error: return code =
0x00010000
Dec 13 15:23:43 demo3 kernel: end_request: I/O error, dev sdn, sector
55676416
Dec 13 15:23:43 demo3 kernel: sd 3:0:0:5: SCSI error: return code =
0x00010000
```

.

StorNext responds to the failure and redirects IO for the LUNs with an active path through c1p0 (LUN-1, 3 and 5) to the PASSIVE path mapped to c0p1.

```
Dec 13 15:23:43 demo3 fsm[5856]: pickpath: I/O path in error [hostid:
3 lun: 1 path </dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:1>]
quiesced for 300 seconds
Dec 13 15:23:43 demo3 fsm[5856]: StorNext File System FSS 'snfs2[0]':
pickpath: I/O path [hostid: 3 lun: 1 path: </dev/disk/by-path/pci-
0000:07:02.1-scsi-0:0:0:1>] quiesced for 300 seconds
```

.

```
Dec 13 15:23:43 demo3 kernel: I/O path in error [hostid: 3 lun: 5
path </dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:5>] quiesced for
300 seconds
```

Path failure and redirect takes ~30s. Three minutes after being disconnected, the c1p0 cable is reattached.

```
.
Dec 13 15:25:36 demo3 kernel: qla2400 0000:07:02.1: LIP reset
occurred (f8e1).
Dec 13 15:25:36 demo3 kernel: qla2400 0000:07:02.1: LIP occurred
(f8e1).
Dec 13 15:25:36 demo3 kernel: qla2400 0000:07:02.1: LOOP UP detected
(4 Gbps).
```

And three minutes later, the paths for the data LUNs are restarted.

```
.
Dec 13 15:28:43 demo3 kernel: scan_disk_path_states: I/O path
[hostid: 3 lun: 5 edev: 0x0x8d0 path </dev/disk/by-path/pci-
0000:07:02.1-scsi-0:0:0:5>] restarted after 300 seconds
Dec 13 15:28:43 demo3 kernel: scan_disk_path_states: I/O path
[hostid: 3 lun: 3 edev: 0x0x8b0 path </dev/disk/by-path/pci-
0000:07:02.1-scsi-0:0:0:3>] restarted after 300 seconds
```

## 2.2. RAID Controller Failure and Recovery

RAID controller failure was tested by issuing a “non-graceful” shutdown to the c1 controller via the c0 controller and removing it from the chassis. (This option is available through the controllers VT100 interface on the “Diagnostics” page.) As with the path failure test, iozone is used to generate IO to the local snfs file system from both cluster nodes. After a 30s lapse, the scsi driver starts reporting IO errors.

```
.
Dec 13 15:32:27 demo3 kernel: qla2400 0000:07:02.1: LOOP DOWN
detected (2).
Dec 13 15:32:58 demo3 kernel: sd 3:0:0:3: SCSI error: return code =
0x00010000
Dec 13 15:32:58 demo3 kernel: end_request: I/O error, dev sdl, sector
17216256
Dec 13 15:32:58 demo3 kernel: sd 3:0:0:5: SCSI error: return code =
0x00010000
Dec 13 15:32:58 demo3 kernel: end_request: I/O error, dev sdn, sector
57413760
```

While the c1 controller is down, StorNext retries the failed paths, logging the following messages when it determines the path is still unavailable

```
.
Dec 13 15:42:58 demo3 kernel: scan_disk_path_states: I/O path
[hostid: 3 lun: 5 edev: 0x0x8d0 path </dev/disk/by-path/pci-
0000:07:02.1-scsi-0:0:0:5>] restarted after 300 seconds
Dec 13 15:42:58 demo3 kernel: sd 3:0:0:5: SCSI error: return code =
0x00010000
Dec 13 15:42:58 demo3 kernel: end_request: I/O error, dev sdn, sector
58027904
Dec 13 15:42:58 demo3 kernel: I/O error on cookie 0xcc33c11 cvfs
error 'IO error' (0x3) os status code 0x3 cvfs disk name '5404_lun5'
hba_id 3 lun 5 blkno 0x6eabb0000 offset 0x3ddb0000 count 0x10000
resid 0x0
Dec 13 15:42:58 demo3 kernel: I/O path in error [hostid: 3 lun: 5
path </dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:5>] quiesced for
300 seconds
```

After running all IO through the c0 controller for ~15min, the c1 controller is reinserted and reboots. On the next probe, StorNext finds the LUN paths UP and restarts them.

```
.
.
Dec 13 15:47:18 demo3 kernel: qla2400 0000:07:02.1: LIP reset
occurred (f8f7).
Dec 13 15:47:18 demo3 kernel: qla2400 0000:07:02.1: LIP (f8f7).
Dec 13 15:47:18 demo3 kernel: qla2400 0000:07:02.1: LOOP UP detected
(4 Gbps).
Dec 13 15:47:58 demo3 kernel: scan_disk_path_states: I/O path
[hostid: 3 lun: 5 edev: 0x0x8d0 path </dev/disk/by-path/pci-
0000:07:02.1-scsi-0:0:0:5>] restarted after 300 seconds
Dec 13 15:47:58 demo3 kernel: scan_disk_path_states: I/O path
[hostid: 3 lun: 3 edev: 0x0x8b0 path </dev/disk/by-path/pci-
0000:07:02.1-scsi-0:0:0:3>] restarted after 300 seconds

Dec 13 15:52:45 demo3 fsm[5856]: pickpath: I/O path [hostid: 3 lun:
1 path </dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:1>] restarted
after 998 seconds
Dec 13 15:52:45 demo3 fsm[5856]: StorNext File System FSS 'snfs2[0]':
pickpath: I/O path [hostid: 3 lun: 1 path: </dev/disk/by-path/pci-
0000:07:02.1-scsi-0:0:0:1>] restarted after 998 seconds
```

### **2.3. Metadata Controller (MDC) Failure and Recovery**

To test MDC failure and recovery we configured the StorNext cluster for failover by editing the *fsnameservers* file on both cluster nodes (refer to Section 7 for file listings). For the MDC failover test the IO load was generated by a StorNext LAN client accessing the snfs file system via the LAN gateway software.

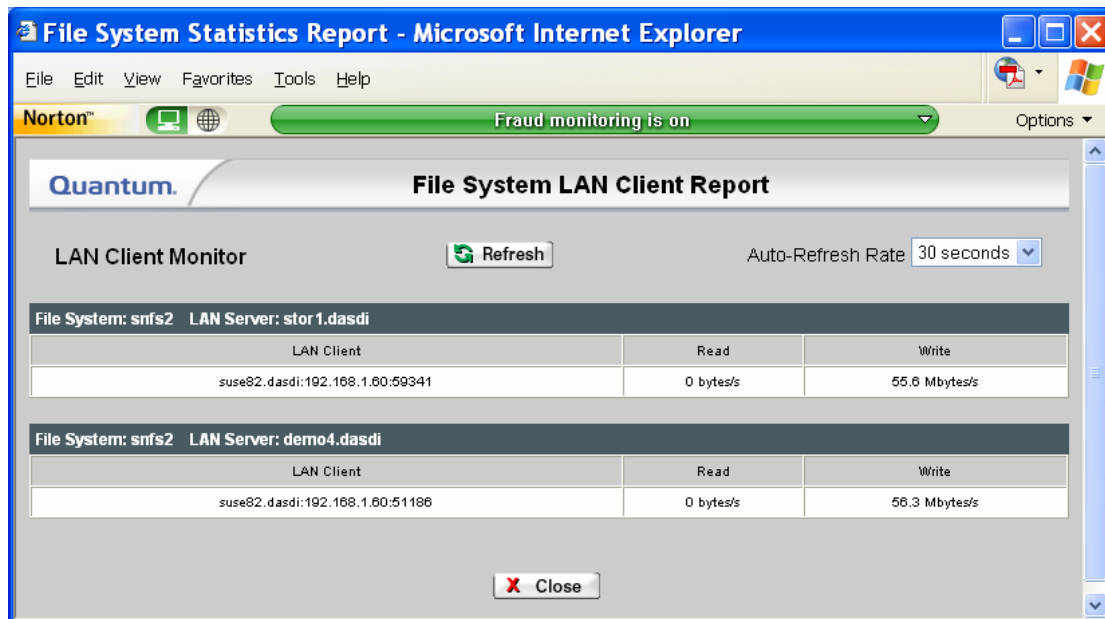
Both cluster nodes mount the file system with the LAN gateway option enabled. From the cluster nodes */etc/fstab*;

```
snfs2 /mnt/snfs2 cvfs rw,diskproxy=server 0 0
```

The LAN client (suse82) mounts the filesystem remotely. From the LAN clients */etc/fstab*;

```
snfs2 /mnt/snfs2 cvfs
    rw,diskproxy=client,buffercache=512,verbose=yes 0 0
```

lozone is started on the LAN client to generate IO into the remote snfs filesystem. The StorNext GUI is used to confirm the LAN client IO is balanced across both LAN gateway servers.



On cluster node demo4 cvadmin is started and confirms that both servers are recognized as MDC capable.

```
demo4:/usr/cvfs/bin # ./cvadmin
StorNext File System Administrator
```

Enter command(s)  
For command help, enter "help" or "?".

List FSS

```
File System Services (* indicates service is in control of FS):
1>*snfs2[0]          located on stor1.dasdi:49266 (pid 5856)
2> snfs2[0]         located on demo4.dasdi:49038 (pid 6976)
```

Select FSM "snfs2"

```
Created           : Fri Nov 30 14:50:15 2007
Active Connections: 3
Fs Block Size     : 4K
Msg Buffer Size    : 4K
Disk Devices      : 6
Stripe Groups     : 6
Fs Blocks         : 6526630656 (24.31 TB)
Fs Blocks Free    : 6505911020 (24.24 TB) (99%)
```

Server demo3 (the host for stor1.dasdi) is shutdown via an init 0. The following snippet is from the /var/log/messages on demo4.dasdi as server demo3 shuts down and the standby MDC on demo4.dasdi takes over.

```
Dec 13 16:12:02 demo4 fsmppm[6953]: PortMapper: Deleting stale mapping
for id 192.168.1.203
Dec 13 16:12:02 demo4 fsmppm[6953]: NSS: election initiated by
192.168.1.224:32775 (id 192.168.1.224) - fsm HB.
Dec 13 16:12:02 demo4 fsmppm[6953]: NSS: Starting vote for FSS snfs2
using 2 voting members: 192.168.1.60, 192.168.1.224.
Dec 13 16:12:04 demo4 fsmppm[6953]: NSS: Vote selected FSS 'snfs2[0]'
at 192.168.1.224:49038 (pid 6976) - attempting activation.
Dec 13 16:12:04 demo4 fsmppm[6953]: PortMapper: Initiating activation
vote for FSS 'snfs2'.
```

```
Dec 13 16:12:04 demo4 fsm[6953]: NSS: Vote call for FSS snfs2 is
inhibited - vote dis-allowed.
Dec 13 16:12:04 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
Branding Arbitration Block (attempt 1) votes 1.
Dec 13 16:12:06 demo4 kernel: Heart Beat Timer for FSS snfs2 on host
192.168.1.210 not detected for 3 seconds. Initiating Fail Over
Protocol.
Dec 13 16:12:06 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Loading Stripe Group "MetaFiles". 18.62 GB.
Dec 13 16:12:06 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Loading Stripe Group "JournFiles". 18.62 GB.
Dec 13 16:12:06 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Loading Stripe Group "DataFiles1". 6.08 TB.
Dec 13 16:12:06 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Stripe Group "MetaFiles" active.
Dec 13 16:12:06 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Loading Stripe Group "DataFiles2". 6.08 TB.
Dec 13 16:12:06 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Stripe Group "JournFiles" active.
Dec 13 16:12:06 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Loading Stripe Group "DataFiles3". 6.08 TB.
Dec 13 16:12:06 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Loading Stripe Group "DataFiles4". 6.08 TB.
Dec 13 16:12:07 demo4 kernel: Reconnecting to FSS 'snfs2'
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: free blocks 1625380786 with 0 blocks currently reserved
for client delayed buffers.Reserved blocks may change with client
activity.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Stripe Group "DataFiles1" active.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: free blocks 1629703101 with 0 blocks currently reserved
for client delayed buffers.Reserved blocks may change with client
activity.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Stripe Group "DataFiles2" active.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: free blocks 1629573693 with 0 blocks currently reserved
for client delayed buffers.Reserved blocks may change with client
activity.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Stripe Group "DataFiles3" active.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: free blocks 1625378688 with 0 blocks currently reserved
for client delayed buffers.Reserved blocks may change with client
activity.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
FSM Alloc: Stripe Group "DataFiles4" active.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
File system 'snfs2' requires UTF8-NFC file names
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
File System Service 'snfs2[0]' now active on host
'demo4.dasdi:49038'.
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
Node [1] [demo4.dasdi:48426] Client Login (active 1).
Dec 13 16:12:21 demo4 kernel: Reconnect successful to FSS 'snfs2' on
host '192.168.1.224'.
Dec 13 16:12:21 demo4 kernel: Using v2 readdir for 'snfs2'
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':
Node [2] [suse82.dasdi:46096] Client Login (active 2).
```



## F54xxE and StorNext File System

```
Dec 13 16:12:21 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':  
Proxy server 192.168.1.224/34004 (pmpport 48922, pmflags 0x0)  
registered with FSM
```

These messages continue; cvadmin from demo4 shows the new state of the cluster. The LAN client IOZONE continues running with all IO going through the surviving LAN gateway on demo4.

```
snadmin (snfs2) > select  
List FSS
```

```
File System Services (* indicates service is in control of FS):  
1>*snfs2[0]          located on demo4.dasdi:49038 (pid 6976)
```

Server demo3 (stor1.dasdi) is rebooted and StorNext is restarted.

```
Dec 13 16:25:50 demo4 fsm[6976]: PortMapper: Added mapping from id  
192.168.1.203 to addr 192.168.1.210  
Dec 13 16:25:51 demo4 fsm[6976]: NSS: Name Server '192.168.1.210'  
(192.168.1.210) port is 32776, revision is 0x0102.  
Dec 13 16:25:51 demo4 fsm[6976]: NSS: Standby FSS 'snfs2[0]' at  
192.168.1.203:43422 (pid 5886) - registered.  
Dec 13 16:25:55 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':  
Node [3] [stor1.dasdi:54148] Client Login (active 3).  
Dec 13 16:25:55 demo4 fsm[6976]: StorNext File System FSS 'snfs2[0]':  
Proxy server 192.168.1.210/52265 (pmpport 45903, pmflags 0x0)  
registered with FSM
```

We check cluster status again with cvadmin.

```
snadmin (snfs2) > select  
List FSS
```

```
File System Services (* indicates service is in control of FS):  
1>*snfs2[0]          located on demo4.dasdi:49038 (pid 6976)  
2> snfs2[0]          located on stor1.dasdi:43422 (pid 5886)
```

StorNext GUI confirms that the LAN gateway on demo3 is again serving data to client suse82.

We fail the MDC back to server demo3 (stor1.dasdi) to complete the test.

```
snadmin (snfs2) > fail snfs2  
Fail over FSS "snfs2"
```

```
FSS 'snfs2' fail over initiated.  
Select FSM "none"
```

```
snadmin > select  
List FSS
```

```
File System Services (* indicates service is in control of FS):  
1>*snfs2[0]          located on stor1.dasdi:43422 (pid 5886)  
2> snfs2[0]          located on demo4.dasdi:37622 (pid 15070)
```

```
snadmin> exit  
demo4:/usr/cvfs/bin #
```

The preceding tests were conducted on clusters using both the F5412E and the F5404E for their RAID storage. Path failovers on both systems typically took on the order of 30s. If faster failovers are required, system administrators can adjust the SCSI driver options. Refer to your FC card driver documentation.

### 3. Array Configuration and Optimization

#### 3.1. F5412E Array Configuration

The F5412E used for testing had 33 drives. Thirty of these drives were SATA2 and 3 were 15K SAS. Two of the SAS drives were used to build an R-1 mirror. From this R-1 mirror we define two 10GB LUNs; LUN-0 for the SNFS metadata and LUN-1 for the SNFS journal.

Of the 30 SATA2 drives eighteen are 750GB and twelve are 250GB. These disks are configured as five R-6 (4+2) arrays. Each of these arrays hosts a single 900GB LUN.

##### 3.1.1. Arrays and LUNs

The diagram below shows how the arrays and LUNs on the F5412E were configured during this test. The different colors correspond to the drives within each array. The numbers are the LUNs hosted within that array.

CH1	2	2	E	0/1
	2	2	E	0/1
	2	2	E	DS
CH2	4	4	3	3
	4	4	3	3
	4	4	3	3
CH3	6	6	5	5
	6	6	5	5
	6	6	5	5



Slots marked with an "E" are empty. All LUNs are visible from all ports.

##### 3.1.2. SNFS Sequential Read/Write Performance

Sequential performance was tested using IOZONE.

```
Single thread test;          /mnt/snfs1/demo1 ./iozone -i0 -i1 -s4g -r64k -T -t1
1 Thread/SG* test;         /mnt/snfs1/demo1 ./iozone -i0 -i1 -s4g -r64k -T -t5
```

	Single Node	Two Nodes
1 Thread*	W-118MB/s R-195MB/s	W-114MB/s R-328MB/s
1 Thread per SG*	W-185MB/s R-392MB/s	W-274MB/s R-562MB/s

\*The data LUNs on this system are R-6 (4+2). StorNext is configured with 1LUN/SG

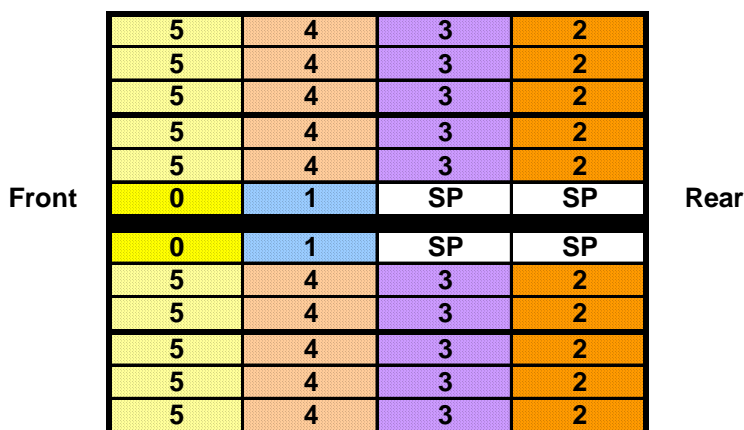
**3.2. F5404E Array Configuration**

The F5404E used for testing was fitted with 48 SATA2 750GB drives. Four drives were used to build two R-1 mirrored arrays. A small (20GB) LUN is hosted on each of these arrays. These are used for StorNext’s metadata and journal.

Forty drives are use to build four R-5 (9+1) arrays. The remaining four drives are assigned as spares.

**3.1.2. Arrays and LUNs**

The diagram below shows how the arrays and LUNs on the F5404E were configured during this test. Different colors correspond to the drives within an array. The numbers are the LUNs hosted within that array.



SATA

Color=Array

Number=LUN

All LUNs are visible on ALL of the RAID controller’s ports (default setting). Read/Write Performance

**3.1.3. SNFS Sequential Read/Write Performance**

Sequential performance was tested using IOZONE.

Single thread test; /mnt/snfs1/demo1 ./iozone -i0 -i1 -s4g -r64k -T -t1  
 1 Thread/SG\* test; /mnt/snfs1/demo1 ./iozone -i0 -i1 -s4g -r64k -T -t4

	Single Node	Two Nodes
1 Thread*	W-135MB/s R-363MB/s	W-322MB/s R-537MB/s
1 Thread per SG*	W-228MB/s R-585MB/s	W-245MB/s R-766MB/s

\*The data LUNs on this system are R-5 (9+1). StorNext is configured with 1LUN/SG.

### **3.3. Array Optimization**

In the F54xxE controllers, an “array” can be divided into multiple LUNs. To the hosts, the LUN is seen as a unique device, but internal to the “array” it is simply a range of LBAs. The workload presented to this array might be sequential in respect to individual LUNs, but if these LUNs are hosted within the same “array”, the actual workload at the drives requires the RAID controller to interleave the IOs from each LUN into the array. The additional head movement to service this IO results in non-sequential operations when it comes to disk accesses. This IO pattern causes additional seek/rotational overhead and therefore the performance of the LUNs will not be the maximum of the array. The performance loss can be substantial when using SATA drives with R-6 due to the additional IO required in the R-6 algorithms. For this reason, one should avoid defining multiple LUNs from a single “array”, especially if the array is parity based (i.e. R-5 and R-6).

### **3.4. Chunk Optimization**

Users are normally suggested to select a chunk size for RAID-5 which matches the predominant IO size from the host. For sequential operation a larger chunk size is suggested. For small random operations a smaller chunk size is suggested. That said, the Xyratex RAID product has been designed and optimized such that large chunk sizes are good for all types of I/O and recommend configurations as such.

A chunk is defined as the amount of drive space on one drive that contributes to a complete array stripe across the array members. The controller supports chunk sizes of 64K, 128K, and 256K.

The above suggestion is made as if the chunk is the unit of access to the drive for all operations, small IO would be burdened by transferring the full chunk size when only part of it suffices the host IO. Large IO would not be efficient if host IO spans multiple chunks that would involve multiple drive accesses and thus reduce performance.

The Xyratex controllers operate in a mode of only accessing drives for the amount of data to service the host IO. This method of service IO requests reduces the performance effects of “poor” stripe size selections.

Large host IO generates large IOs to the drives. The controller will use read ahead optimization and write coalescing to align the host IOs to the underlying array structure. In this manner the maximum potential performance of the array in sequential IO is achieved.

Advantages can be gained for small host IO with small chunk sizes, but these require the IO to align to the underlying stripe structure. This is not typical for most host IO streams. Users with small random IO streams should consider if this is an optimization.

The largest chunk size of 256K therefore gives the lowest overheads for commands issued in sequential operation. This should be selected for all sequential IO streams. Additionally with no penalty for additional drive transfers implemented by the controller in random IO, random performance is not compromised with this setting.

StorNext’s configuration file can be optimized to take advantage of the RAID arrays ability to efficiently do “full stripe” writes. For best throughput, match StorNext’s *StripeBreadth* parameter to the disk arrays “full-stripe” write size (chunk\_size x number\_of\_data\_drives) in the array.

### **3.5. Write Cache Optimization**

The write cache can be optimized in a number of ways. The amount of cache that is allocated to a given volume can be controlled. This allows system administrators to allocate cache resources between arrays and therefore prioritise one array over another due to the amount of cache resources it is assigned.

The cache, when constrained, moves from write back mode to write through mode as cache resources are consumed. Large sequential IO will consume those resources quickly and moves the workflow into write through mode for this type of IO. This optimizes performance of the controller to the incoming workflow. If small IO is interleaved with large IO, this offers a good all round solution to this workflow.

The default setting for all arrays is 16MB of write-back cache. This number is chosen as it has been shown to give the best mix of write-back and write-through behavior for general purpose servers. Unless the IO stream is all random or all sequential IO, this setting will act as a good compromise for most mixed IO streams.

The write cache is dynamically reconfigurable and all changes are implemented immediately. Users can therefore change settings without affecting access to the arrays.

### **3.6 Read Cache Optimization**

The read cache is adaptive in nature. If enabled, the read ahead algorithm only is implemented if the controller detects "sequentialness" in the incoming host IO flow. The controller implements a complex searching algorithm that looks back through the workflow, more than just next command to detect sequential IO. If "sequentialness" is happening between multiple streams, read ahead is preserved, unlike in basic adaptive techniques.

The read cache can also be constrained to prioritise accesses of cache resources between arrays in a similar manner to the write cache.

The constraining abilities allow the users to segregate cache for read/write and between individual arrays. In this manner users can gain fine control of the cache resources. If required, the user can allow all arrays to have equal performance by setting the write cache to MAXIMUM. At this setting, the controller will dynamically arbitrate available cache resources amongst the arrays.

The read cache is dynamically reconfigurable and all changes are implemented immediately. Users can therefore change settings without affecting accesses to arrays.

## 4. Performance Counters

The following section briefly introduces the performance counters of the F54xxE controllers. Users are encouraged to use this feature as it allows them to understand more about the incoming workflow and therefore what potential optimization can be done within the controller to improve performance.

The Statistics are broken into 3 main areas

- **Workload**
- **Cache**
- **Array**

Each of these sections will be discussed

### 4.1. Workload

The statistics of the incoming commands are collected by the controller and can be presented in a number of ways to understand the workflow. The raw data collected includes

- **Number of commands**
- **Data transferred**
- **Access size**
- **Alignment**

This data can be collected

- **Per controller**
- **Per Port**
- **Per array**
- **Read or write or both**

Users should use these counters to check the total workload and how that relates to the maximum performance capabilities of the controllers and arrays as documented earlier in the white paper. Users should consider the nature of the IO and if it is random or sequential in nature. The optimization discussed in section 3 should then be considered to see if any of these apply to the nature of the workload.

The User should also then look at individual ports and arrays to check if these are operating at their respective maximums or not. This will allow the user to load balance the incoming host IO stream as suggested in the workflow optimization section.

### 4.2. Cache

The statistics collected by the controllers are separate for read and write caches.

#### Read Statistics

The following statistics are collected

- **Hit rate**
- **Read ahead efficiency**

#### Write Statistics

The following statistics are collected

- **Cluster rate**
- **Partial cluster rate**
- **Full stripe write rate**
- **Cluster count**

A cluster is defined as a region of sequential commands, this may equate into a chunk or stripe depending on the amount of clustered commands.

The user can use these statistics to increase or decrease the amount of cache allocated to specific volumes. Insufficient cache may cause good hit/cluster rates, but the controller may be operating less optimally. Users should look to allocate enough cache to gain high hit rates, but no more than necessary.

### **4.3. Array**

A range of additional statistics are gathered that relate to the performance of the arrays, these include

- **Cache full events**
- **Initiator Queue full signalled**
- **Read & write sequences to the arrays**

The users should investigate these issues to find out the reason for these issues occurring. If they can be reduced in the host application, incremental performance can be obtained from the controller.

## 5. Conclusions

The Xyratex F54xx series controllers have been shown to work well with Quantum's StorNext file system software. The file system remained online while being subjected to multiple types of hardware failures (path failure, disk failure, RAID controller failure, MDC failure) during a series of comprehensive tests performed from 11/15/2007 to 12/14/2007.

Linux persistent naming (/dev/disk/by-path) was used within cvlabels and cvpaths files to define unique paths to each LUN for failover and load-balancing. With this configuration, and the Xyratex controllers' ability to present the same LUN on multiple ports, failover was handled completely within the context of the StorNext software. No 3<sup>rd</sup> party MPIO or Linux dm/multipath facilities are needed.

Both RAID-5 and RAID-6 arrays were successfully tested with large (>2TB) LUNs. The maximum sequential write performance using the StorNext file system was 322MB/s with eight streams. Maximum sequential read performance from StorNext was 766MB/s. These are quite close to the design spec of 330MB/s and 830MB/s when using SATA drives.

Xyratex will continue to improve performance capability of the controller over the life of the product. This will be done without impacting integrity of the controller. Xyratex has demonstrated highly innovative solutions to many performance problems. These provide no compromises to basic integrity and availability of the controller.

## 6. Appendix

### **a. StorNext SNFS Control Files**

The following files are from StorNext cluster snfs2.

#### **i. snfs2.cfg**

```
#
*****
*****
# A global section for defining file system-wide parameters.
#
# For Explanations of Values in this file see the following:
#
# UNIX Users:      man cvfs_config
#
# Windows Users:  Start > Programs >
#                  StorNext File System > Help >
#                  Configuration File Format
#
#
*****
*****

ABMFreeLimit          No
AllocationStrategy    Round
# BufferCacheSize      64M      # Default is 32MB
DataMigration         No        # SNMS Managed File Systems Only
# DataMigrationThreadPoolSize 128 # Default is 8 (Managed only)
Debug                 0x0
FileLocks             No
ForceStripeAlignment  Yes
FsBlockSize           4K
GlobalSuperUser       Yes      # Set to Yes for SNMS Managed File
Systems
InodeCacheSize        16K      # 800-1000 bytes each, default 8K
InodeExpandMin        32K
InodeExpandInc        128K
InodeExpandMax        8M
JournalSize           4M
MaxConnections        32
# ReservedSpace Yes    # NO: Slows small I/Os. Causes
fragmentation.
MaxLogs               4
MaxLogSize            16M
# OpHangLimitSecs     300      # Default is 180 secs
Quotas                No
ThreadPoolSize        32      # Default is 16, 512 KB memory per
thread
UnixDirectoryCreationModeOnWindows  0755
UnixFileCreationModeOnWindows        0644
UnixIdFabricationOnWindows            No
UnixNobodyGidOnWindows                60001
UnixNobodyUidOnWindows                 60001
WindowsSecurity                         Yes
```



**F54xxE and StorNext File System**

```

#
*****
*****
# A disktype section for defining disk hardware parameters.
#
*****
*****

[DiskType MetaDrive]
Sectors 39057375
SectorSize 512

[DiskType JournalDrive]
Sectors 39057375
SectorSize 512

[DiskType DataDrive1]
Sectors 13053265887

#
*****
*****
# A disk section for defining disks in the hardware configuration.
#
*****
*****

[Disk 5404_lun0]
Type MetaDrive
Status UP

[Disk 5404_lun1]
Type JournalDrive
Status UP

[Disk 5404_lun2]
Type DataDrive1
Status UP

[Disk 5404_lun3]
Type DataDrive1
Status UP

[Disk 5404_lun4]
Type DataDrive1
Status UP

[Disk 5404_lun5]
Type DataDrive1
Status UP

#
*****
*****
# A stripe section for defining stripe groups.
#
*****
*****

```



## F54xxE and StorNext File System

```
[StripeGroup MetaFiles]
StripeBreadth 256K
Type Regular
MetaData Yes
Status UP
Node 5404_lun0 0
Write Enabled
Exclusive Yes
Read Enabled
```

```
[StripeGroup JournFiles]
StripeBreadth 256K
Type Regular
Status Up
Exclusive Yes
Write Enabled
Node 5404_lun1 0
Read Enabled
Journal Yes
```

```
[StripeGroup DataFiles1]
StripeBreadth 576K
Type Regular
Status Up
Node 5404_lun2 0
Write Enabled
Read Enabled
```

```
[StripeGroup DataFiles2]
StripeBreadth 576K
Type Regular
Status Up
Write Enabled
Node 5404_lun3 0
Read Enabled
```

```
[StripeGroup DataFiles3]
StripeBreadth 576K
Type Regular
Status Up
Node 5404_lun4 0
Write Enabled
Read Enabled
```

```
[StripeGroup DataFiles4]
Status Up
Type Regular
Read Enabled
Write Enabled
StripeBreadth 576K
Node 5404_lun5 0
```

### **ii. cvlabels**

```
#This system is attached to the 5404 ports c0p0 and c1p0
5404_lun0 /dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:0 # host 6
lun 0 sectors 39057375 sector_size 512 inquiry [XYRATEX F5404E
] serial 60050CC000F00902000000000000000008
```

```
5404_lun1 /dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:1 # host 6
lun 1 sectors 39057375 sector_size 512 inquiry [XYRATEX F5404E
] serial 60050CC000F0090200000000000000009
5404_lun2 /dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:2 # host 6
lun 2 sectors 13053265887 sector_size 512 inquiry [XYRATEX F5404E
] serial 60050CC000F009020000000000000000A
5404_lun3 /dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:3 # host 6
lun 3 sectors 13053265887 sector_size 512 inquiry [XYRATEX F5404E
] serial 60050CC000F009020000000000000000B
5404_lun4 /dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:4 # host 6
lun 4 sectors 13053265887 sector_size 512 inquiry [XYRATEX F5404E
] serial 60050CC000F009020000000000000000C
5404_lun5 /dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:5 # host 6
lun 5 sectors 13053265887 sector_size 512 inquiry [XYRATEX F5404E
] serial 60050CC000F009020000000000000000D
```

**iii. cvpaths**

```
#1st HBA for even LUNs, attached to C0P0
device=/dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:0 hba=2 lun=0
verify=5404_lun0 usage=Active
device=/dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:1 hba=2 lun=1
verify=5404_lun1 usage=Passive
device=/dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:2 hba=2 lun=2
verify=5404_lun2 usage=Active
device=/dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:3 hba=2 lun=3
verify=5404_lun3 usage=Passive
device=/dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:4 hba=2 lun=4
verify=5404_lun4 usage=Active
device=/dev/disk/by-path/pci-0000:07:02.0-scsi-0:0:0:5 hba=2 lun=5
verify=5404_lun5 usage=Passive
#2nd HBA for odd LUNs, attached to C1P0
device=/dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:0 hba=3 lun=0
verify=5404_lun0 usage=Passive
device=/dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:1 hba=3 lun=1
verify=5404_lun1 usage=Active
device=/dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:2 hba=3 lun=2
verify=5404_lun2 usage=Passive
device=/dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:3 hba=3 lun=3
verify=5404_lun3 usage=Active
device=/dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:4 hba=3 lun=4
verify=5404_lun4 usage=Passive
device=/dev/disk/by-path/pci-0000:07:02.1-scsi-0:0:0:5 hba=3 lun=5
verify=5404_lun5 usage=Active
#
```

**iv. fsmlist**

```
snfs2
```

**v. fsnameservers**

```
192.168.1.210
192.168.1.224
```

**vi. dpservers**

```

# =====
# Disk Proxy Server Configuration
# -----

# =====
# Tunable Parameters
# =====

# Un-comment and modify any of the following lines to override the
default
# values for tunable parameters.
#tcp_window_size_kb 64
#transfer_buffer_size_kb 256
#transfer_buffer_count 16
#server_buffer_count 8
#daemon_threads 8

# =====
# Interface Configuration
# =====

# One or more interfaces must be configured for Disk Proxy Server
I/O.

# Un-comment the line below to allow Disk Proxy Server I/O on
interface 'eth1'.
#interface eth1

# Un-comment the line below to allow Disk Proxy Server I/O on
interface 'eth3'.
interface eth3

# Un-comment the line below to allow Disk Proxy Server I/O on
interface 'bond0'.
#interface bond0

```