



CATE Storage and Backup Engineering  
CATE CitiSystems Engineered Systems  
Citigroup Architecture and Technology Engineering

---

# ScaleIO Environments on Tornado++ Hardware

**CSBE-15004-BP**

**Best Practice**

**1.1**

©Citigroup Inc. 2015, All Rights Reserved.

This document contains material confidential to Citigroup. The contents are protected by specific employee and contractor agreements regarding confidentiality. Reproduction for distribution outside of Citigroup is prohibited.

## 1 Version Control

| Version | Date         | Author                                                               | Reviewed                     | Comments                                                     |
|---------|--------------|----------------------------------------------------------------------|------------------------------|--------------------------------------------------------------|
| 0.1     | 27-May-2015  | Chris Howarth,<br>Chris Clayton,<br>Scott Torgerson                  | Bob Small,<br>Ludwig Derodel | Peer                                                         |
| 0.5     | 2-June-2015  | Chris Howarth,<br>Chris Clayton,<br>Scott Torgerson,<br>Eliot Wilson | Global Stakeholders          | RFT                                                          |
| 1.0     | 22-June-2015 | Chris Howarth,<br>Chris Clayton,<br>Scott Torgerson,<br>Eliot Wilson | Global Stakeholders          | Release to Production                                        |
| 1.1     | 17-Aug-2015  | Chris Howarth,<br>Chris Clayton,<br>Scott Torgerson,<br>Eliot Wilson |                              | Updated Performance expectations and QoS limit Best Practice |
|         |              |                                                                      |                              |                                                              |
|         |              |                                                                      |                              |                                                              |
|         |              |                                                                      |                              |                                                              |
|         |              |                                                                      |                              |                                                              |
|         |              |                                                                      |                              |                                                              |

## 2 Legal Notice

Information contained herein is for Internal Use and may be used only for business purposes authorized by Citigroup.

CITI, CITIBANK, CITICORP, and CITIGROUP are trademarks and service marks of Citicorp and are used and registered throughout the world. Citibank, Citicorp, Citigroup and their subsidiaries also claim rights in certain other trademarks and service marks contained in these pages.

## 3 Contact

Please address any comments or questions regarding the information contained in this document via email to the [\\*CATE Global Engineering Support – ScaleIO](#) distribution list.

## 4 Latest Versions

It is recommended that the latest version of the document be used from the CSBE website at the following URL:  
<http://catecollaboration.citigroup.net/domains/platstor/gdse/products/default.aspx>.  
Newer versions may include certification on additional platforms, or comments on issues discovered after deployment.

## 5 Table Of Contents

|           |                                                                          |           |
|-----------|--------------------------------------------------------------------------|-----------|
| <b>1</b>  | <b>VERSION CONTROL .....</b>                                             | <b>2</b>  |
| <b>2</b>  | <b>LEGAL NOTICE.....</b>                                                 | <b>3</b>  |
| <b>3</b>  | <b>CONTACT.....</b>                                                      | <b>3</b>  |
| <b>4</b>  | <b>LATEST VERSIONS .....</b>                                             | <b>3</b>  |
| <b>5</b>  | <b>TABLE OF CONTENTS .....</b>                                           | <b>4</b>  |
| <b>6</b>  | <b>TABLE OF FIGURES.....</b>                                             | <b>5</b>  |
| <b>7</b>  | <b>INTRODUCTION.....</b>                                                 | <b>6</b>  |
| <b>8</b>  | <b>APPLICABLE ENVIRONMENTS.....</b>                                      | <b>6</b>  |
| <b>9</b>  | <b>BEST PRACTICES .....</b>                                              | <b>7</b>  |
| 9.1       | TORNADO++ HARDWARE CONFIGURATION.....                                    | 7         |
| 9.1.1     | <i>ScaleIO on Tornado++ Hardware .....</i>                               | <i>8</i>  |
| 9.1.2     | <i>MDM and Tie-Breaker Placement in Racks .....</i>                      | <i>9</i>  |
| 9.1.3     | <i>Expansion of existing ScaleIO clusters.....</i>                       | <i>9</i>  |
| 9.1.4     | <i>Protection Domains on Tornado++ .....</i>                             | <i>10</i> |
| 9.2       | FAULT SETS ON TORNADO++.....                                             | 10        |
| 9.3       | POOLS ON TORNADO++ .....                                                 | 11        |
| 9.3.1     | <i>Data Layout on Pools .....</i>                                        | <i>11</i> |
| 9.3.2     | <i>Pool Usage.....</i>                                                   | <i>13</i> |
| 9.3.3     | <i>Performance expectations.....</i>                                     | <i>13</i> |
| 9.3.4     | <i>Throttling volume IO (ScaleIO QoS) .....</i>                          | <i>14</i> |
| 9.4       | POOL SPARE CAPACITY & MAXIMUM POOL USAGE .....                           | 15        |
| 9.4.1     | <i>VMware volume provisioning .....</i>                                  | <i>18</i> |
| 9.5       | RAM READ CACHE SETTING .....                                             | 19        |
| 9.5.1     | <i>RAM Read Cache Write Handling Mode.....</i>                           | <i>19</i> |
| 9.6       | SCALEIO NETWORK CONFIGURATION ON TORNADO++.....                          | 21        |
| 9.6.1     | <i>Private ScaleIO Cluster Network for SDS/MDM/TB Configuration.....</i> | <i>22</i> |
| 9.6.2     | <i>SDS Client Facing Network Configuration .....</i>                     | <i>22</i> |
| 9.6.3     | <i>SDC Network Configuration.....</i>                                    | <i>22</i> |
| 9.6.4     | <i>Network Failure Recovery Processes .....</i>                          | <i>23</i> |
| 9.7       | INSTALLATION AND CONFIGURATION .....                                     | 24        |
| 9.7.1     | <i>Hardware Configuration .....</i>                                      | <i>24</i> |
| 9.7.2     | <i>ScaleIO Node Build.....</i>                                           | <i>24</i> |
| 9.7.3     | <i>ScaleIO Cluster Configuration .....</i>                               | <i>26</i> |
| 9.8       | VMWARE STANDARDS.....                                                    | 26        |
| 9.8.1     | <i>VMWare Installation.....</i>                                          | <i>26</i> |
| 9.8.2     | <i>VMWare Provisioning .....</i>                                         | <i>26</i> |
| <b>10</b> | <b>REFERENCES.....</b>                                                   | <b>27</b> |
| <b>11</b> | <b>APPENDIX A - REBUILD/REBALANCE TIME ESTIMATES .....</b>               | <b>28</b> |

|                                                                                |           |
|--------------------------------------------------------------------------------|-----------|
| <b>12 APPENDIX B - SCALEIO ON TORNADO++ : HARDWARE MTBF .....</b>              | <b>31</b> |
| 12.1 COMPARISON WITH TRADITIONAL STORAGE ARRAYS .....                          | 32        |
| <b>13 APPENDIX C – NETWORKING RELATED PACKAGES REQUIRED FOR TORNADO++.....</b> | <b>32</b> |
| 13.1 NETWORK CARRIER DETECTION FAILURE .....                                   | 33        |
| 13.2 SCALEIO SDS ROUTING CONFIGURATION PACKAGE .....                           | 34        |

## 6 Table of Figures

|                                                                   |    |
|-------------------------------------------------------------------|----|
| Figure 1 - Typical Tornado++ Hardware Configuration .....         | 7  |
| Figure 2 - SDS Types and Location in Tornado++ .....              | 8  |
| Figure 3 - MDM and Tie-Breaker Placement in Tornado++ Racks ..... | 9  |
| Figure 4 - Fault Set layout in Tornado++ .....                    | 10 |
| Figure 5 - Tornado++ Storage Pools.....                           | 11 |
| Figure 6 - Pool Layout on SSDs.....                               | 12 |

## 7 Introduction

EMC ScaleIO is software that creates a SAN by aggregating servers' local direct-attached storage (DAS) to form globally shared block storage accessible over IP. The lightweight ScaleIO software components are installed on the application hosts and inter-communicate via a standard LAN to handle the application I/O requests sent to ScaleIO block volumes. This decentralized block I/O flow combined with a distributed, sliced volume layout results in parallelism which allows it to scale too many nodes (servers).

A ScaleIO volume is a block device that is exposed to one or more hosts (much like a traditional SAN LUN). ScaleIO breaks each volume into a large number of 1MB data chunks which are spread across the servers and disks within the ScaleIO cluster. Each volume also has a corresponding mirror on different nodes and servers in the cluster. This layout goes a long way in eliminating hot spots across the cluster and allows for the scaling of the overall I/O performance of the system through the addition of nodes and or disks. Furthermore, this layout enables a single application that is accessing a single volume to use the full IOPs of all the cluster's disks (much like XIV storage arrays).

Traditional storage systems typically combine system software with commodity hardware (which is comparable to application server hardware) to provide enterprise-grade resilience. ScaleIO provides similar enterprise-grade resilience by running the storage software directly on application servers. Designed for fault tolerance and high availability, ScaleIO handles all types of failures including failures of media, connectivity, nodes, software interruptions, and more.

The ScaleIO software features distributed auto-healing processes that can overcome media and node failures without requiring administrator involvement. Dynamic and elastic, ScaleIO enables administrators to add or remove nodes and capacity on the fly. The software immediately responds to changes by rebalancing the storage distribution to achieve a layout that optimally suits the new configuration. All performance and redistributions on ScaleIO can be monitored and observed on the ScaleIO dashboard which is a simple Java applet.

Because ScaleIO is hardware agnostic, the software works with various types of storage including magnetic disks, solid-state disks, and PCIe flash cards.

## 8 Applicable Environments

*This document ONLY applies to ScaleIO environments running on Tornado++ hardware.* Any other ScaleIO deployments should consult the [ScaleIO R1.32.0 Solutions Guide](#) that covers general best practices for hardware configurations other than Tornado++.

## 9 Best Practices

All ScaleIO best practices shown in the following sections on apply to ScaleIO environments running on standard Tornado++ hardware configurations. All best practices for ScaleIO running on general type hardware environments can be found in the [EMC ScaleIO R1.32.0 Solutions Guide](#).

### 9.1 Tornado++ Hardware configuration

The standard Tornado++ Pod is made up of (2) Compute Blocks and (2) Storage Blocks as shown below. ScaleIO environments running on Tornado++ hardware will only use the (4) racks that make up the two Storage Blocks.

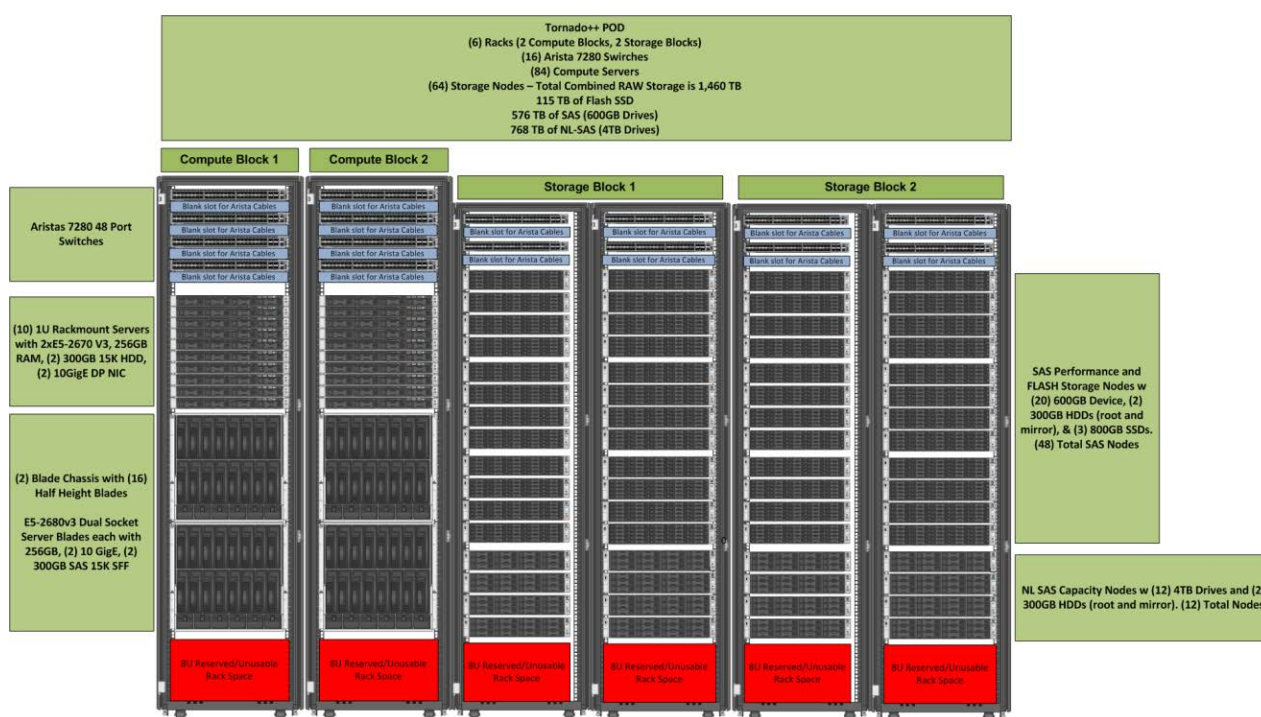


Figure 1 - Typical Tornado++ Hardware Configuration

**Note:** This document will not take into account or consider partial Tornado++ environments. All best practices for ScaleIO running on Tornado++ hardware will only be considered in (4) rack increments. Administrators desiring to deploy ScaleIO environments other than that (4) racks of Tornado++ hardware should contact the authors of this document for further discussion.

### 9.1.1 ScaleIO on Tornado++ Hardware

When ScaleIO environments are placed on Tornado++ hardware it would only utilize the two Storage Blocks. ScaleIO will use all (48) of the Nodes containing SSD and SAS drives but only use (8) of the (16) high capacity nodes containing the NL-SAS drives as shown in the diagram below.

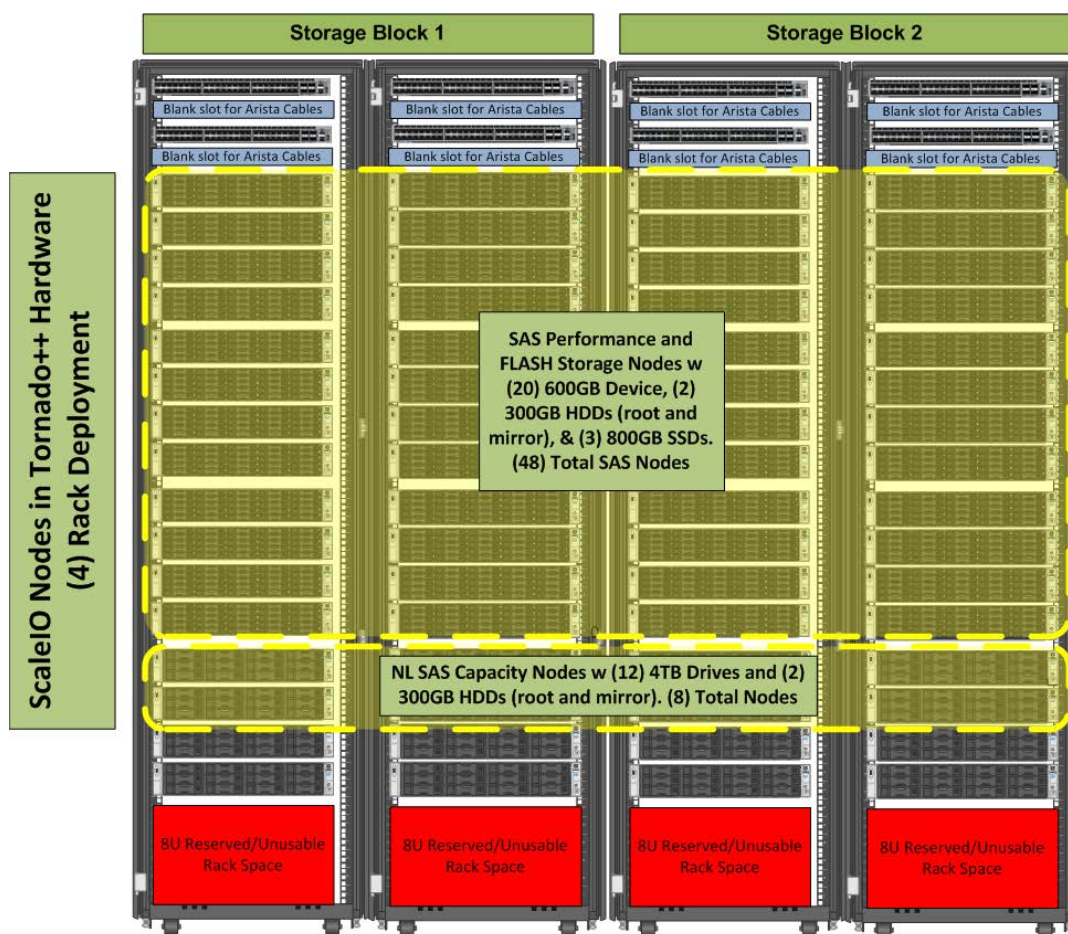


Figure 2 - SDS Types and Location in Tornado++

The ScaleIO environment would have the following capacity for each Tier.

| Tier           | Drive size | Total Number of Drives | Total Raw Capacity (TB) | Max Usable Capacity* (TB) |
|----------------|------------|------------------------|-------------------------|---------------------------|
| SSD            | 800 GB     | 144                    | 104.7                   | 52.4                      |
| SAS (per pool) | 600 GB     | 960                    | 174.3                   | 87.2                      |
| NLSAS          | 4 TB       | 96                     | 349.2                   | 174.6                     |

\*Note that the actual usable will be smaller than this once the spare capacity has been subtracted (see below).

The total raw capacity of ScaleIO running on Tornado++ hardware will be ~977TB. When factoring in mirroring and spare capacity, the usable total space across all pools is ~445.3TB, although at the time of writing some additional space also needs to be left unused in each pool to permit rebuilds to successfully complete (see section on Pool Spare Capacity & maximum pool usage).

The capacity by pool type is shown below.

- SSD Pool – 50 TB usable
- SAS Pools – 243 TB usable in total (81TB per pool)
- NL-SAS Pool – 152 TB usable

### 9.1.2 MDM and Tie-Breaker Placement in Racks

Each cabinet in the Tornado++ is completely independent except that each pair of cabinets shares the same pair of TOR network switches. However due to the resilient network design described later in this document, there is no single point of failure (SPOF). Consequently the ScaleIO Primary MDM, Secondary MDM, Tie-Breaker and Standby MDM will each be configured on the first node in each rack, respectively as shown below:

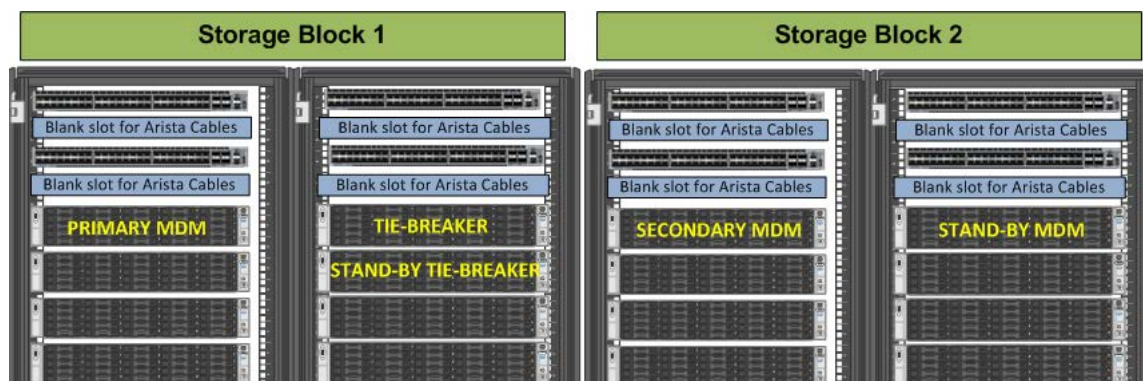


Figure 3 - MDM and Tie-Breaker Placement in Tornado++ Racks

### 9.1.3 Expansion of existing ScaleIO clusters

Best Practice on expanding ScaleIO capacity on the Tornado++ hardware platform will be provided in a future edition of this document.

However, it should be noted for planning purposes that CATE recommends that: ScaleIO clusters be built with Tornado++ servers purchased from a single vendor and that HP and Lenova server are not mixed in the same cluster due to variations on performance and configuration of their respective disk controllers.

### 9.1.4 Protection Domains on Tornado++

A Protection Domain is a set of SDSs. Each SDS belongs to one (and only one) Protection Domain. Thus, by definition, each Protection Domain is a unique set of SDSs. For the Tornado++ configuration a single protection domain should be used. *A single protection domain should be created containing all SDS nodes from all (4) racks.*

### 9.2 Fault Sets on Tornado++

A Fault Set is a logical entity that contains a group of SDSs within a Protection Domain that have a higher probability of failing as a single unit; for example if they are all located in the same rack. ScaleIO creates two copies of the data. Each copy will be placed in a different fault set.

When ScaleIO is configured on the Tornado++ hardware, *each of the racks and the SDS nodes in that rack will be its own fault set as shown below.*

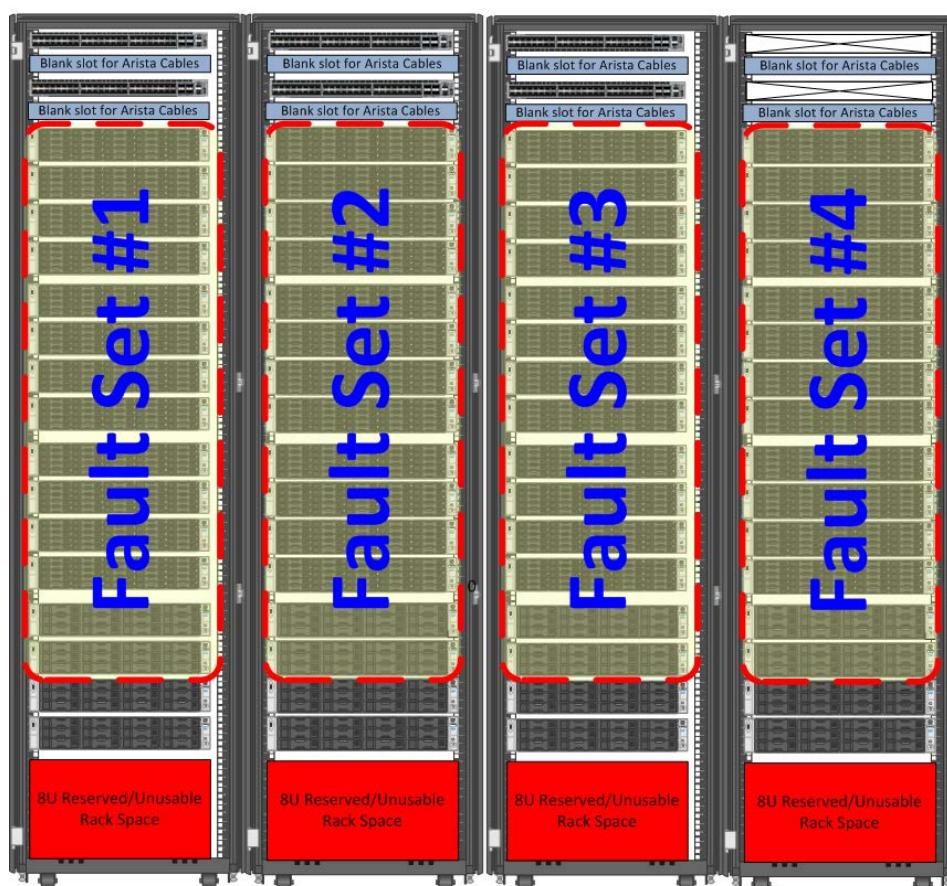


Figure 4 - Fault Set layout in Tornado++

### 9.3 Pools on Tornado++

The following type pools should be created when running ScaleIO on Tornado++ hardware.

- SSD Pool – Using the (3) SSD drives in each of the SAS SDS Nodes
- (3) SAS Pools – Using (20) drives in (16) SAS SDS Nodes.
- NL-SAS Pool – Using (12) devices in each of the (8) NL-SAS SDS Nodes.

**Note:** When the (3) SAS Pools are created, careful attention should be paid to how the data is distributed across the (48) SDS Nodes. The [Data Layout on Pools](#) section of this document will discuss this in more detail.

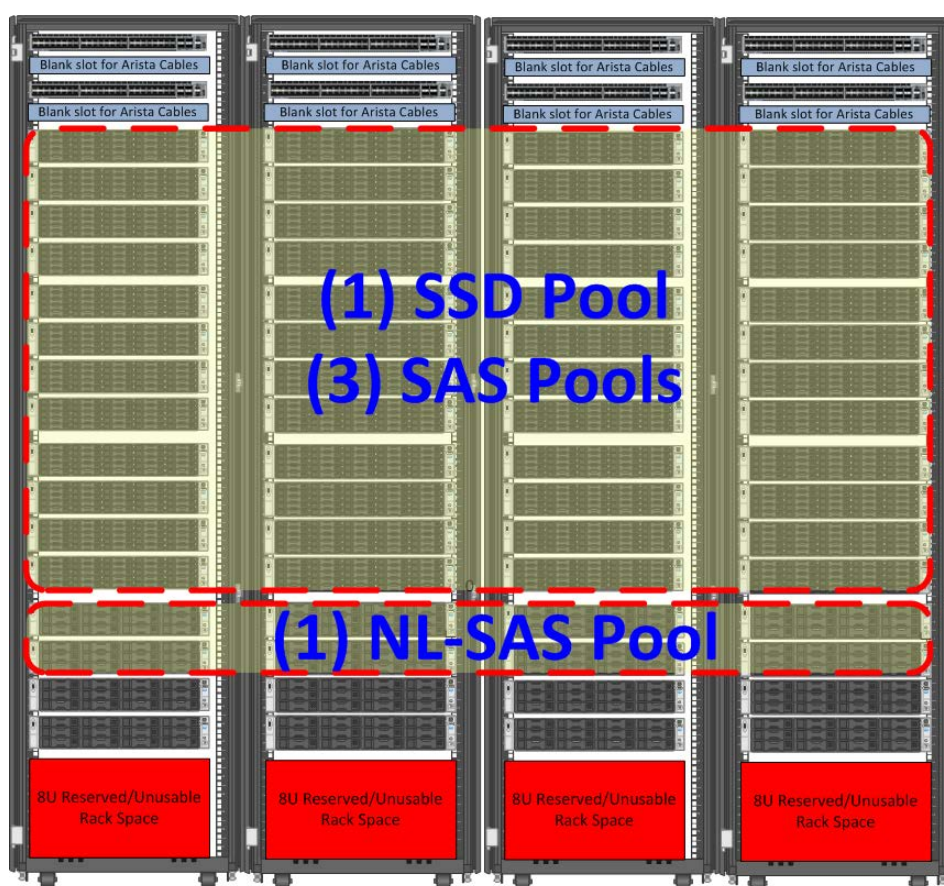


Figure 5 - Tornado++ Storage Pools

#### 9.3.1 Data Layout on Pools

When creating both the SSD and NL-SAS pools, every disk in each of the respective SDS node types should be used for that pool and SDS type. However, when creating the (3) SAS pools, the following list of requirements should be followed.

- Each of the (3) SAS pools should be spread across 320 SAS drives.
- Each SAS pool should only span (16) SAS SDS nodes.

**Note:** Failure to take the above requirements into account can result in the spare calculations being incorrect.

When accounting for the above items, the following diagram should be used when laying out all ScaleIO Storage Pools on Tornado++ SDS nodes.

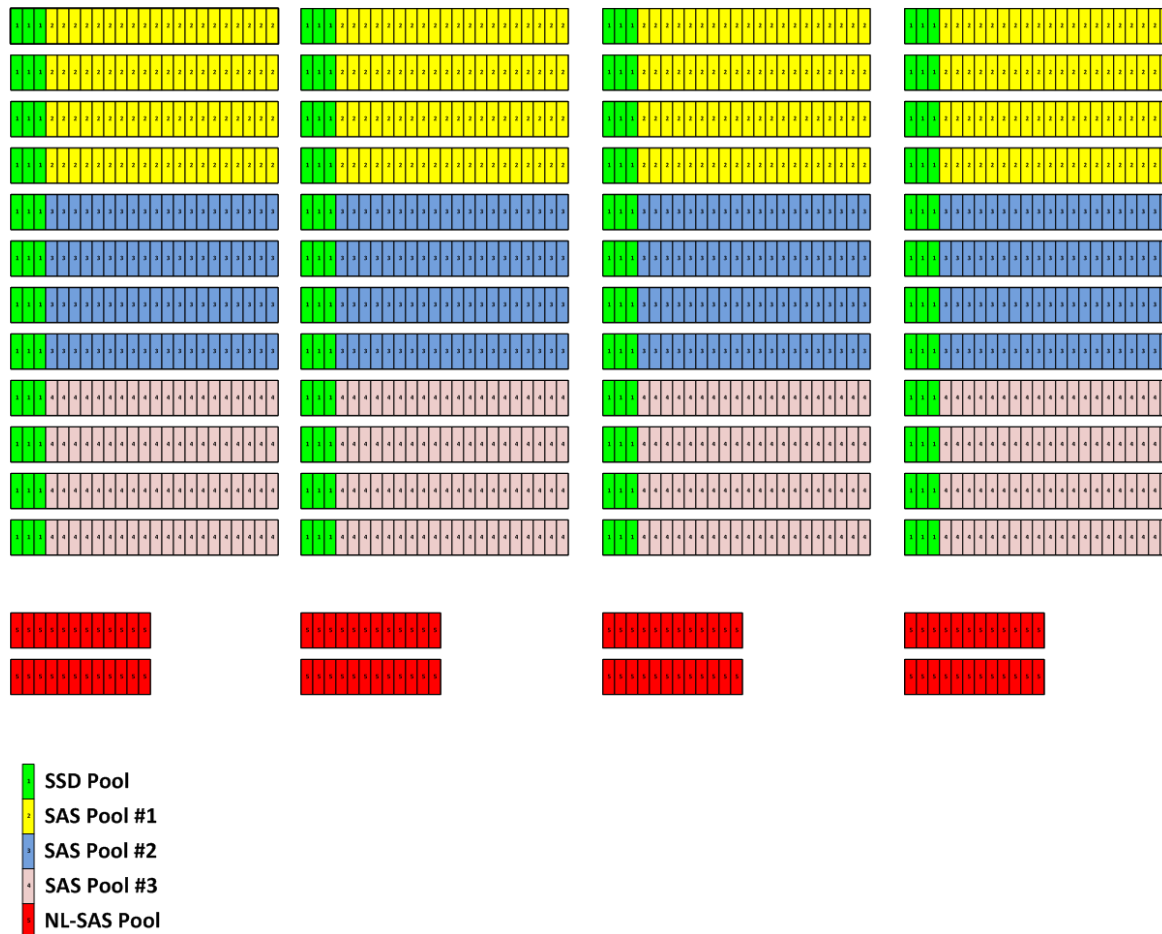


Figure 6 - Pool Layout on SSDs

### 9.3.2 Pool Usage

The following table shows how the storage pools are to be used. Note that there will be no over commitment of storage resources until ScaleIO thin-provisioning concerns are resolved. At this point thin-provisioning may be used for physical server use cases, but for VMware cases thin provisioning will be managed VMware not ScaleIO.

| Pool        | Usage             | IO Throttling by ScaleIO | IO Throttling by VMware (SIOC) | Over commitment of storage resources |
|-------------|-------------------|--------------------------|--------------------------------|--------------------------------------|
| SAS Pool 1  | VMware            | No                       | Yes                            | No                                   |
| SAS Pool 2  | VMware            | No                       | Yes                            | No                                   |
| SAS Pool 3  | TBD               |                          |                                | No                                   |
| SSD Pool    | VMware & Physical | Yes                      | Yes                            | No                                   |
| NL-SAS Pool | VMware & Physical | Yes                      | Yes                            | No                                   |

Note that if SAS pool 3 is used for Physical hosts (only) there will be no IO throttling by VMware (SIOC).

**Note:** no inter-pool data mobility will be available until 2016.

### 9.3.3 Performance expectations

The following table shows the estimated capabilities of the SSD, SAS and NL-SAS pools, based on the underlying disk capabilities.

| ScaleIO on Tornado++ |              |               |               |                     |                                                |                                                       |
|----------------------|--------------|---------------|---------------|---------------------|------------------------------------------------|-------------------------------------------------------|
| Tier                 | Racks in pod | Servers /rack | Disks /server | Total disks per pod | Max POD IOPS Host IOPS (OLTP2HW-type workload) | Sustained Host IOPS/usable TB (OLTP2HW-type workload) |
| 800GB SSD            | 4            | 12            | 3             | 144                 | 120,000                                        | 25,000                                                |
| 600GB SAS            | 4            | 12            | 20            | 960                 | 126,000<br>(42,000 per pool)                   | 600                                                   |
| 4TB NL-SAS           | 4            | 2*            | 12            | 96                  | 11,333                                         | 85                                                    |

\* Assumes only 2 of the 4 NL-SAS nodes in each rack is used for ScaleIO capacity-optimized tier

### 9.3.4 Throttling volume IO (ScaleIO QoS)

#### 9.3.4.1 Summary of recommended ScaleIO QoS limits

| Pool type           | IOPS/TB limit | bandwidth/TB limit |
|---------------------|---------------|--------------------|
| SSD pool volumes    | 50,000        | 2,344              |
| SAS pool volumes    | 1,200         | 450                |
| NL-SAS pool volumes | 170           | 64                 |

**Per TB ScaleIO QoS limits by pool type**

| Pool type           | VMware volume size (GB) | Volume IOPS limit | Volume Bandwidth limit |
|---------------------|-------------------------|-------------------|------------------------|
| SSD pool volumes    | 2,560                   | 125,000           | 5,860                  |
| SAS pool volumes    | 5,120                   | 6,000             | 2,250                  |
| NL-SAS pool volumes | 11,264                  | 1,870             | 704                    |

**Per Volume ScaleIO QoS limits for VMware standard size volumes**

**Important note:** the above limits are actually “per LUN per SDC” (see below) and not just “per LUN”. Hence, \*if\* multiple SDCs are able to access the same volume simultaneously (e.g. via a clustered file system), then the above limits on IO to a volume can be exceeded.

#### 9.3.4.2 Throttling Volume I/O on NL-SAS Pools

The NL-SAS tier is intended for cold (inactive) data. Hence, it is backed by low-cost, high-capacity and very-low performance disks. This tier can only support around 30 sustained random IOPS per TB. Excessive IO to this tier can impact the performance of that tier for all users. Hence it is recommended that all volumes provisioned from the NL-SAS tier are throttled, with 2X over-subscription of IOPS i.e. to 60 IOPS/TB, and also have a bandwidth limit set (optimized for 128KB IOs, such as those used by Netbackup).

The limits must be set on a volume-by-volume basis for each of the SDC(s) to which it is mapped.

```
# sclcli --set_sdc_volume_limit --volume_name <VOLUME-NAME> --sdc_name
<SDN-NAME> --limit_iops <IOPS LIMIT> --limit_bandwidth <BANDWIDTH
LIMIT>
```

e.g.

```
# sclcli --set_sdc_volume_limit --volume_name slowvol --sdc_name
myserver --limit_iops 1870 --limit_bandwidth 700 --i_am_sure
```

Successfully modified limits from SDC myserver to volume slowvol:IOPS  
limit set to 1870.Bandwidth limit set to 700 MB/sec

This command sets limits to the IOPs and bandwidth that one SDC generates for the specified volume. The example above is for an 11264GB volume. The value for iops and bandwidth varies with the volume size and can be calculated for an arbitrary-sized volume as follows:

**NL-SAS pool volume limits**

```
limit_iops = volume size (in GB) * 170 / 1024
limit_bandwidth = volume size (in GB) * 64 / 1024
```

### 9.3.4.3 Throttling Volume I/O on SSD & SAS Pools

Volumes created from the SSD tier on Tornado++ also need to be throttled for completely different reasons. The Tornado++ SSD have a very high specification and hence it is currently recommended to throttle them to ensure that they do not consume all the IOPS capability of the SDS nodes and impact access to the SAS tier which is serviced by the same set of SDS nodes.

SAS tiers should also be throttled to avoid “noisy neighbour” issues.

The “per TB” throttle values for the SSD and SAS pools, and the “per volume” values for standard-sized VMware volumes are shown in the table at the start of this section. Command syntax is identical to the NL-SAS example above (just with different limit values).

The value for iops and bandwidth varies with the volume size and should be calculated for an arbitrary-sized volume as follows:

**SSD pool volume limits**

```
limit_iops = volume size (in GB) * 50000 / 1024
limit_bandwidth = volume size (in GB) * 2344 / 1024
```

**SAS pool volume limits**

```
limit_iops = volume size (in GB) * 1200 / 1024
limit_bandwidth = volume size (in GB) * 64 / 1024
```

## 9.4 Pool Spare Capacity & maximum pool usage

To ensure data protection during server failures, ScaleIO reserves 10% of the capacity (of each pool) by default, not allowing it to be used for volume allocation. To ensure full system protection during the event of an SDS node failure, *ScaleIO administrators must make sure that the spare capacity is at least equal*

to the amount of capacity in a single SDS node along with enough capacity to lose a few extra disks.

As all SDS nodes should contain equal capacity, it is recommended to set the capacity value to at least  $1/N + 2$  disks in each SSD and SAS pools, of the total capacity where N is the number of SDS nodes in that pool. For the NL-SAS pool only a single node of spare capacity should be figured (1/N).

The best practice and calculation for each ScaleIO pool on Tornado++ SDS nodes is shown below.

**Note:** All spare capacity figures are rounded up to the next percentage.

**SSD Pool spare percentage = 4%**

5 (spare drives) / 144 (total drives) = 3.5% rounded up to 4%

**SAS Pool spare percentage = 7%** (For each of the 3 SAS pools)

22 (spare drives) / 320 (total drives) = 6.8% rounded up to 7%

**NL-SAS Pool spare percentage = 13%**

12 (spare drives) / 96 (total drives) = 12.5% rounded up to 13%

Note, however, that during Tornado++ testing, it was found that the above spare percentages were insufficient to allow rebuilds to complete when a node is lost. The issue has been raised with EMC, whom have been tasked with providing a process to calculate the additional spare/unused capacity needed in an environment with multiple fault sets. Citi testing has found that the following pool usage values must not be exceeded in a 4-rack Tornado++ configuration to ensure that rebuild can complete:

| Pool type            | Pool size (in usable GB) | Maximum allowed pool "In Use" (GB) | Maximum allowed pool "In use" (%) | Spare capacity to be set (%) |
|----------------------|--------------------------|------------------------------------|-----------------------------------|------------------------------|
| <b>SSD pool</b>      | 53580                    | 48750                              | 91%                               | 4%                           |
| <b>Each SAS pool</b> | 89260                    | 72300                              | 81%                               | 7%                           |
| <b>NL-SAS pool</b>   | 178799                   | 135900                             | 76%                               | 13%                          |

**Pool maximum utilization limits and spare capacity settings**

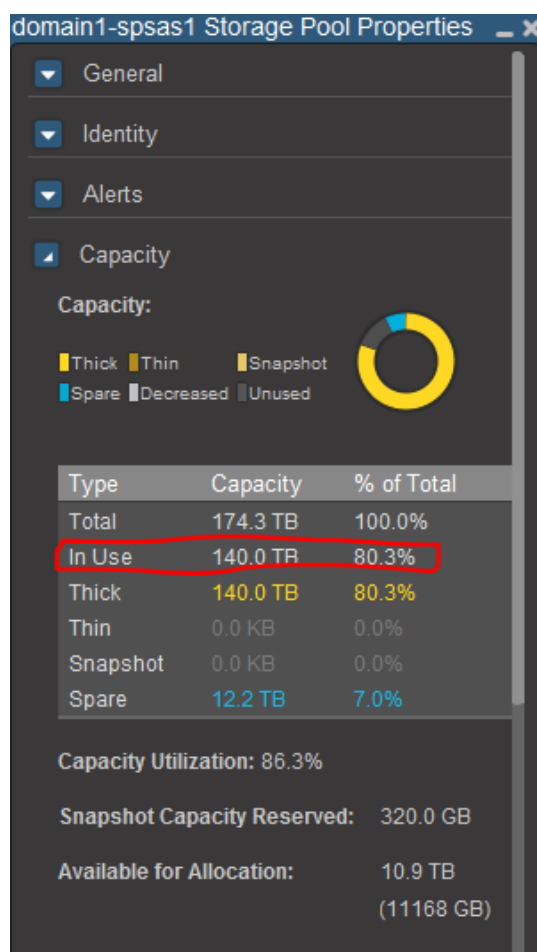
The current pool utilization (GB) can be obtained using the following command.

```
# scli --query_storage_pool --storage_pool_name (pool-name> --
protection_domain_name <domain-name> | grep in-use
```

For example,

```
# scli --query_storage_pool --storage_pool_name domain1-spsas1 --
protection_domain_name domain1 | grep in-use
1.8 TB (1818 GB) in-use capacity
```

The current pool “In use” percentage can be obtained via the dashboard only, by looking at the “In Use” figure highlighted below.



### Determining the pool “In use” percentage

The maximum number of standard VMware LUNs that comply with this restriction are listed in the next subsection. **Do not create more LUNs than specified below and do not exceed the maximum allowed pool utilization percentages listed in the table above.**

To ensure that the levels of maximum utilization do not result in critical threshold alerts, or excessive instances of high threshold alerts, the following command should be run to increase the high and critical alerts level to higher values than the default values.

```
# scli --set_capacity_alerts_threshold --
capacity_high_threshold 87 --capacity_critical_threshold 95
```

The alerts thresholds are calculated from each Storage Pool capacity after deducting the defined amount of spare capacity. So with that in mind, the “Capacity Utilization” that each pool has once all the VMware LUNs are created is:

NL-SAS: 86.9% (currently showing as a medium alert as above high threshold)  
 SAS: 86.3% (currently showing as a medium alert as above high threshold)  
 SSD: 94.6 (currently showing as a high alert as above critical threshold)

Sadly, this is a single setting for all the pools, but with this change, then the cluster will only have the SSD pool showing up as being above the high threshold.

#### 9.4.1 VMware volume provisioning

The maximum number of VMware volume that should be provisioned out of each of the ScaleIO Tornado++ pools is specified in the following table.

| Pool type            | VMware standard volume size (GB) | Maximum number of such volumes | QoS IOPS Limit to be set per volume * | QoS bandwidth Limit to be set per volume * |
|----------------------|----------------------------------|--------------------------------|---------------------------------------|--------------------------------------------|
| <b>SSD pool</b>      | 2560                             | 19                             | 125,000                               | 4,000                                      |
| <b>Each SAS pool</b> | 5120                             | 14                             | 6,000                                 | 2,350                                      |
| <b>NL-SAS pool</b>   | 11264                            | 12                             | 1,870                                 | 700                                        |

#### Maximum number of standard VMware volumes per pool

\* See the section on [Throttling volume IO \(ScaleIO QoS\)](#)

## 9.5 RAM Read Cache Setting

In a Tornado++ environment, each SDS node will have 64GB of memory. The RAM Read Cache setting in this case would be 65% of the total memory or:

$64\text{GB} * 65\% = 41.6 \text{ GB}$  (rounded up to 42GB).

To allow Tornado++ SDS nodes to actually use 65% of the total memory, the following entries must be placed in the `/opt/emc/scaleio/sds/cfg/conf.txt` file.

**Note:** It would be best to make the edits to this file when the SDS rpm packages are initially installed on the SDS.

```
rcache_allowed_high_memory_percentage=80
rcache_allowed_low_memory_percentage=80
rcache_allowed_high_memory_precentage=80
rcache_allowed_low_memory_precentage=80
numa_affinity=1000
numa_memory_affinity=0
```

If the above edits were no made when the SDS rpm package was installed or prior to the cluster being configured, the SDS service would have to be restarted on each SDS node where the config file is changed.

```
# /opt/emc/scaleio/sds/bin/delete_service.sh
# /opt/emc/scaleio/sds/bin/create_service.sh
```

**Note:** If the SDS service needs to be restarted on each SDS node after the cluster is configured, it should be done serially and any rebuild/rebalance should be allowed to complete prior to restarting services on the next SDS node.

Employing RAM cache is not beneficial with SSD devices. RAM cache should only be enabled for HDDs pools. Hence, the RAM cache should be enabled for the SDS nodes, but then disabled for SSD-based storage pools. The following command should be run on any SSD pool deployed on Tornado++ hardware.

```
# scli --set_rmcache_usage --protection_domain_name <domain> --
storage_pool_name <name-of-flash-pool> --dont_use_rmcache
```

### 9.5.1 RAM Read Cache Write Handling Mode

The `rmcache_write_handling_mode` parameter determines if the writes of the Storage Pools are stored in the SDS RAM read cache or if it to be passed

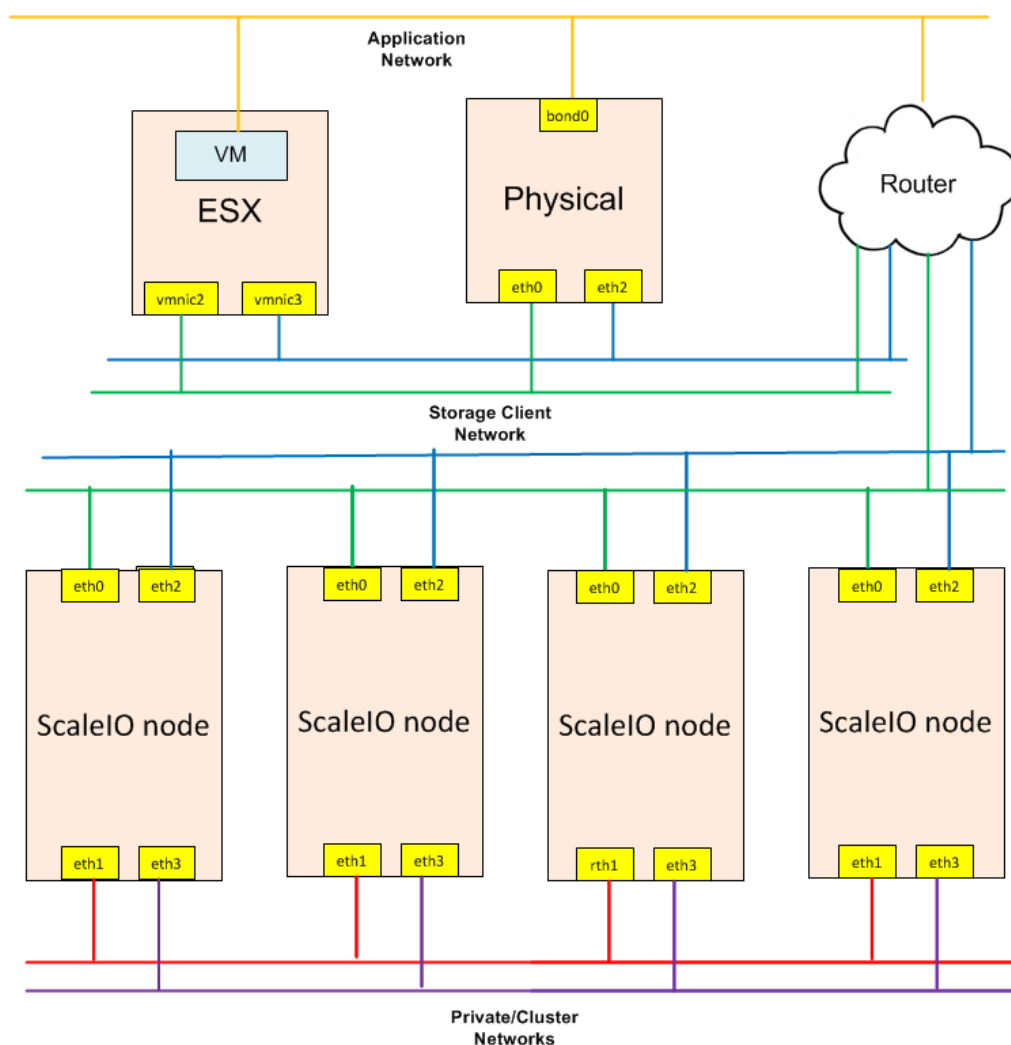
through to the storage. For all Tornado++ hardware environments this parameter should be left at the default value which is *cached*.

## 9.6 ScaleIO Network Configuration on Tornado++

The ScaleIO network design has been designed to make best use of the available network bandwidth, whilst providing no single point of failure. It departs from the traditional model of using pairs of bonded network adapters in an active-passive configuration. Full details of the merits and reasoning behind this can be found in the following document:

[http://ewiki.nam.nsroot.net/w/images/a/a6/ScaleIO\\_Network\\_Options\\_v1.6.pdf](http://ewiki.nam.nsroot.net/w/images/a/a6/ScaleIO_Network_Options_v1.6.pdf)

The following diagram provides a logical view of the Tornado++ network:



At the bottom of the figure the ScaleIO nodes are shown. Two independent networks are to be configured for the private network (shown in purple and red) and two independent networks are configured for the public/client facing network (green and blue).

The following sections describe the configuration of the networks in more detail:

### 9.6.1 Private ScaleIO Cluster Network for SDS/MDM/TB Configuration

There are two storage networks for each ScaleIO node. Nodes within the same rack will be on the same subnet. IP address blocks will be defined for each network (one for the red network and one for the purple). Nodes in a different rack will use a different subnet. Hence static routes will need to be defined for each node as follows:

- o One static route entry for the purple network for nodes in different racks
- o One static route entry for the red network

### 9.6.2 SDS Client Facing Network Configuration

There are two public/client networks for each node. In a similar fashion to the private network, nodes within the same rack will be on the same subnet. IP address blocks will be defined for each network (one for the blue network and one for the green). Nodes in a different rack will use a different subnet.

Static routes will need to be defined for the front end network between racks as per the storage network example. In addition, the client side network connection of the SDS needs to contain a default gateway address. As there are two networks on the client side the default gateway requires the ability to switch network in the event of a network failure or switch maintenance. Hence the following is required:

- The front end default gateway
- The front end SDS traffic

Packages are provided to configure the static routing for both the client and cluster networks for the ScaleIO nodes. These are described in the cluster installation and configuration in section 9.7. and further detail of these packages is described in section 13.

### 9.6.3 SDC Network Configuration

#### 9.6.3.1 Physical Hosts

Each physical host in T++ consuming ScaleIO storage will have two adapters on the same networks as the ScaleIO Storage node client network (shown as green and blue above). This will be separate from their front end client network (shown in yellow).

The hosts will require static routes defining on these networks in a similar fashion to those defined in 9.6.1.

Note that the initial ScaleIO implementation will only have ESXi clients and no physical hosts

#### 9.6.3.2 ESXi Hosts

Each ESXi host will have two 10Gbit network adapters. Each connection should be assigned to a standard virtual switch and VMkernel port group with a unique subnet.

The hosts will require static routes defining on these networks in a similar fashion to those defined in 9.6.1.

The routing addresses will be provided by Network Engineering and will persist across reboots.

Fill details can be found in the VMware Configuration Guide for T++

#### 9.6.4 Network Failure Recovery Processes

The failure of any network component in the T++ POD environment results in no immediate action required. All ScaleIO cluster components including the SDCs, SDSs, MDMs and TB are aware of the multiple access paths to communicate with each other and are able to re-route the network traffic accordingly.

## 9.7 Installation and Configuration

A full description of the installation and configuration of ScaleIO is provided in the Solutions Guide. In addition, specifically for the Tornado++ configuration, the following applies:

### 9.7.1 Hardware Configuration

Prior to shipment, the hardware vendor has been requested to perform the following tasks:

- The disk array controller will have pre-configured the internal disks as follows:
  - o Two drives mirrored for the root O.S.
  - o All remaining SSD/SAS/NL-SAS drives are configured with a one to one physical to logical mapping
- The array cache controller settings to be sanity checked. In particular the ratio of read cache to write cache should be 25:75

### 9.7.2 ScaleIO Node Build

Each ScaleIO nodes will have a standard Citi SOE build. After this, the following additional steps need to be undertaken prior to handover to the CATE Engineering team for testing and then a second time following the server rebuild prior to handover to the CTI team building and supporting ScaleIO:

1. Ensure that each server has the correct number of disks and that the drive bays used are identical across all servers:
  - SAS/SSD servers
    - o 2 x 300GB SAS drives mirrored for root (/dev/sda)
    - o 3 x 800GB SSD drives (sdb → sdb)
    - o 20 x 600GB SAS drives (sde onwards)
  - NL-SAS server
    - o 2 x 300GB SAS drives mirrored for root (/dev/sda)
    - o 12 x 4TB NL-SAS drives (sdb onwards)
2. Configure the four network adapters on each node with an IP address and with jumbo frames. The adapter names will be:

```
<hostname>  
<hostname>-eth1  
<hostname>-eth2  
<hostname>-eth3
```

An example is shown below. Note the MTU is set to 8800. Obviously the IPADDR and NETMASK values will vary:

```
# cat /etc/sysconfig/network-scripts/ifcfg-eth2
DEVICE=eth2
HWADDR=8C:DC:D4:AA:7A:BC
TYPE=Ethernet
UUID=f5ea68fa-930f-4fbd-8f75-7bf27a79e828
IPADDR=10.116.19.197
NETMASK=255.255.255.240
MTU=8800
ONBOOT=yes
NM_CONTROLLED=yes
BOOTPROTO=none
```

Note that the static routes should not be defined as these are automatically set in the steps later below

3. It is currently necessary to install the HP Array Controller CLI as this is currently RFT and not yet part of the standard build. To do so add the following yum repo entry:

```
# cat /etc/yum.repos.d/soe-testing.repo
[soe6products-testing]
name=SOE Linux 6 products including VTMs - Testing
baseurl=http://guegs3/SOELinux/repos/testing/soe6products-$basearch
enabled=0
gpgcheck=0
```

Next install the CLI software (hpssacli is required which is part of this bundle)

```
# yum --enablerepo=soe6products-testing install
hp_monitoring_agents
```

4. Add an entry for the yum repo containing the storage packages:

```
# cat /etc/yum.repos.d/soe-fssprod.repo
[soe6fssproducts]
name=SOE File System and Storage Products
baseurl=http://rh6lab764.nam.nsroot.net/SOELinux/repos/testing/soe6fssp
roducts-$basearch
enabled=1
gpgcheck=0
```

5. Install the package to monitor the network carrier

```
# yum install carrier-mon
```

## 6. Install the package to configure the routing for all the adapters

```
# yum install sdsroute  
  
# /opt/SDSroute/bin/sds_cfgroute  
INFO: Using latest config file from CSBE website 1.0.2  
INFO: REBOOT is required to activate changes
```

Reboot the server for the routing to take effect, then verify the routing is correct:

```
# /opt/SDSroute/bin/sds_verifyroute  
INFO: Using latest config file from CSBE website 1.0.2  
Verification Successful.
```

## 7. Install the ssh key trust package to be used to ease ScaleIO installation

```
# yum install tmpkey
```

**NOTE:** The tmpkey package is only to be used to assist with the initial configuration and **must** be removed after the cluster has been configured.

### 9.7.3 ScaleIO Cluster Configuration

The ScaleIO cluster to be installed as per the ScaleIO Solutions Guide. CTI are using custom scripts for the ScaleIO installation

## 9.8 VMWare Standards

### 9.8.1 VMWare Installation

The SDC installation for VMWare is described in the ScaleIO Solutions guide.

### 9.8.2 VMWare Provisioning

VMWare Tornado++ pool usage and the use of I/O throttling is described in the general section on [Pool usage](#).

## 10 References

| Document                                              | Location                                                                                                                                                                                                                                                                                                                                                              |
|-------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CATE Scale-out commodity block storage position paper | <a href="https://catecollaboration.citigroup.net/domains/platstor/gdse/products/Eval/Scale_Out_Block_storage_position_paper_V1.0.pdf">https://catecollaboration.citigroup.net/domains/platstor/gdse/products/Eval/Scale_Out_Block_storage_position_paper_V1.0.pdf</a>                                                                                                 |
| ScaleIO Product Page                                  | <a href="https://catecollaboration.citigroup.net/domains/platstor/gdse/products/techstdshmpgs/emcscaleIO.aspx">https://catecollaboration.citigroup.net/domains/platstor/gdse/products/techstdshmpgs/emcscaleIO.aspx</a>                                                                                                                                               |
| ScaleIO Standard Build Page                           | <a href="https://catecollaboration.citigroup.net/domains/platstor/gdse/products/platforms/EMC%20ScaleIO/ScaleIO_csb.aspx">https://catecollaboration.citigroup.net/domains/platstor/gdse/products/platforms/EMC%20ScaleIO/ScaleIO_csb.aspx</a>                                                                                                                         |
| ScaleIO v1.31.1277 GNAST Solutions Guide.             | <a href="https://catecollaboration.citigroup.net/domains/platstor/gdse/products/platforms/EMC%20ScaleIO/GNAST%20Design%20and%20Configuration%20Guide%20-%20EMC%20ScaleIO%20R1.31.pdf">https://catecollaboration.citigroup.net/domains/platstor/gdse/products/platforms/EMC%20ScaleIO/GNAST%20Design%20and%20Configuration%20Guide%20-%20EMC%20ScaleIO%20R1.31.pdf</a> |
| EMC ScaleIO v1.31.1277 Users Guide                    | <a href="https://catecollaboration.citigroup.net/domains/platstor/gdse/products/platforms/EMC%20ScaleIO/EMC%20ScaleIO%20Users%20Guide%20v1.31.pdf">https://catecollaboration.citigroup.net/domains/platstor/gdse/products/platforms/EMC%20ScaleIO/EMC%20ScaleIO%20Users%20Guide%20v1.31.pdf</a>                                                                       |
| ScaleIO Security Model                                | <a href="http://ewiki.nam.nsroot.net/w/images/f/f6/ScaleIO_Security_Model_071014.pdf">http://ewiki.nam.nsroot.net/w/images/f/f6/ScaleIO_Security_Model_071014.pdf</a>                                                                                                                                                                                                 |
| VMWare VSI Storage                                    | <a href="https://catecollaboration.citigroup.net/domains/platstor/gdse/products/Designs/CSBE-11079-DD.pdf">https://catecollaboration.citigroup.net/domains/platstor/gdse/products/Designs/CSBE-11079-DD.pdf</a>                                                                                                                                                       |
|                                                       |                                                                                                                                                                                                                                                                                                                                                                       |

## 11 Appendix A - Rebuild/rebalance time estimates

The following table and graphs are intended to provide metrics to allow the estimation of how long a rebuild or rebalance operation is likely to take when an SDS fails , or is re-added to a cluster.

Assumptions:

- The amount of storage allocated is as per the CATE recommendations for VMware specified in the section in this document on [VMware volume provisioning](#).
- All of this provisioned storage has been written to, and hence needs to be re-protected. This is a worse case situation, as in practice some of the storage will not have been written to when the failure occurs, and hence the amount of storage needing to be rebuilt will be less than in the table below.
- Host IO is not high, and hence not competing with the rebuild for spindle IOPS.

| Activity                                                  | SSD pool                                            | SAS pool                                           | NL-SAS pool                                                                                                                                                                      |
|-----------------------------------------------------------|-----------------------------------------------------|----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Rebuild after SDS loss (many → many)</b>               | Duration: 3-4mins for 2.0TB<br><br>Average 9.8GB/s  | Duration: 30 mins for 8.7TB<br><br>Average 4.8GB/s | Duration: 6h55m for 32.2TB<br><br>Average: 1.3GB/s<br><br>Max. sustained rebuild rate: 1.7GB/s (much less towards end)                                                           |
| <b>Subsequent rebalance activity* (many → many)</b>       | Duration: 1-2 mins for 0.5TB<br><br>Average 5.7GB/s | Duration: 18 mins for 2.7TB<br><br>Average 2.6GB/s | Duration: 81 mins for 4.7TB<br><br>Average: 1.0GB/s<br><br>Max. sustained rebalance rate: 1.2GB/s (much less towards end)                                                        |
| <b>Rebalance when SDS returns to cluster (many → one)</b> | Duration: 114 mins for 2.0TB **                     | Duration: 102 mins for 8.7TB                       | Est. duration: 11+ hours for 32.2TB.<br><br>(Est. of duration assumes an average of 1.0GB/s throughput, then adding 20% to duration due to throughput fall off towards the end). |

**Observed rebuild & rebalance durations (or rates) for Tornado++ \*\*\***

\* Rebalance activity does not start until all outstanding rebuild activities across all pools are completed.

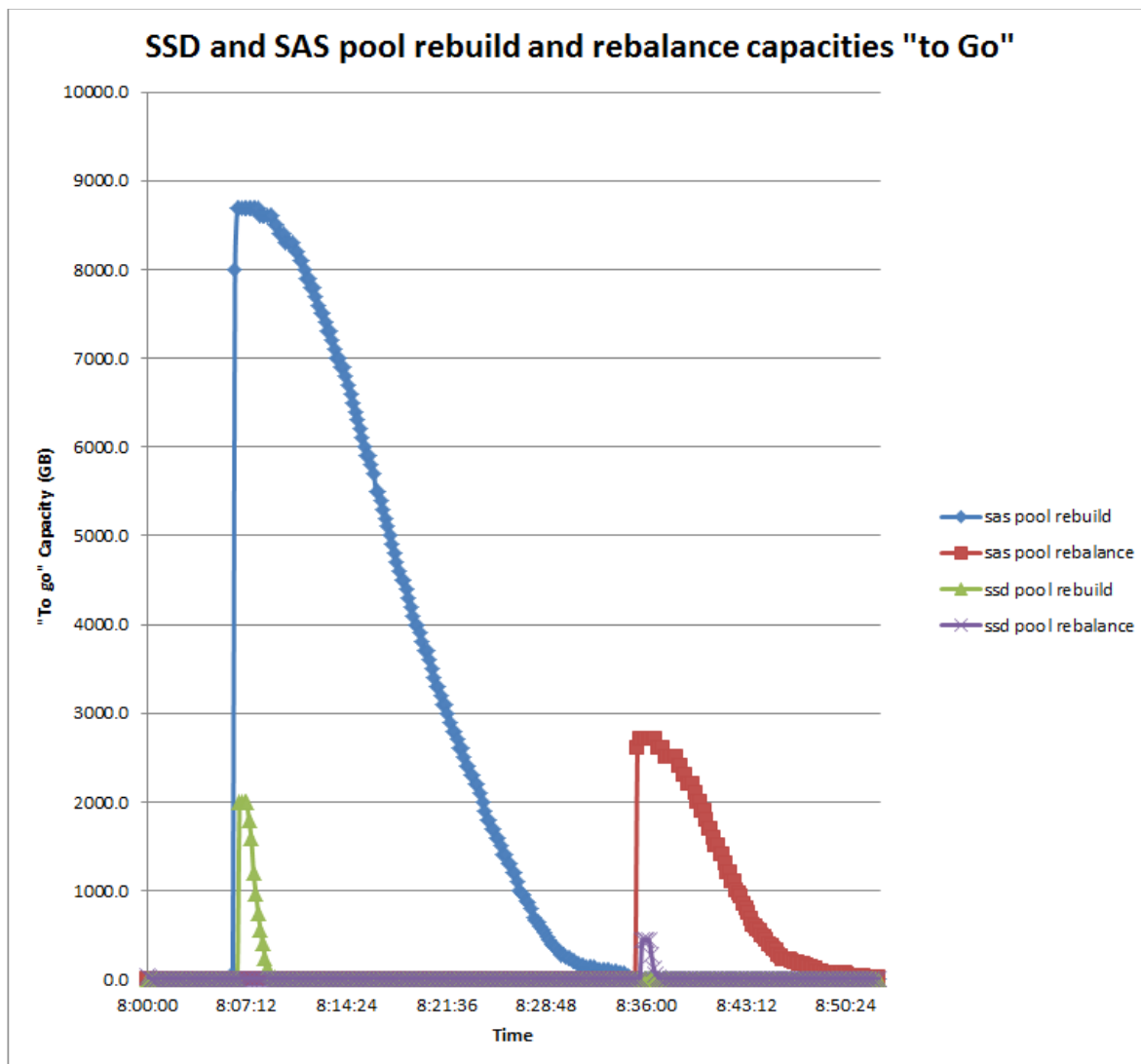
\*\* SSD rebalance is very slow whilst SAS rebalance is occurring, but then completes quickly once that rebalance has completed.

\*\*\* Based on R1.311 with CATE recommendations for pool utilization for VMware deployments, allocated data being fully written to (worst case) and low host IO during rebuild/rebalance (best case)

For example, the following graph shows how the rebuild and rebalance progress with time following the failure of a Tornado++ SDS that supports both the SSD

and one of the SAS pools. The “To go” capacity represents the amount of data that needs to be re-protected (rebuilt) or rebalanced.

SSD and SAS pool rebuilds (green and blue curves respectively) commence immediately after the SDS failure, and rapidly reprotect the degraded capacity. Once both rebuilds complete (“To go” capacity reduces to zero for both pools) then the SSD and SAS pool rebalancing commences (purple and red curves respectively).

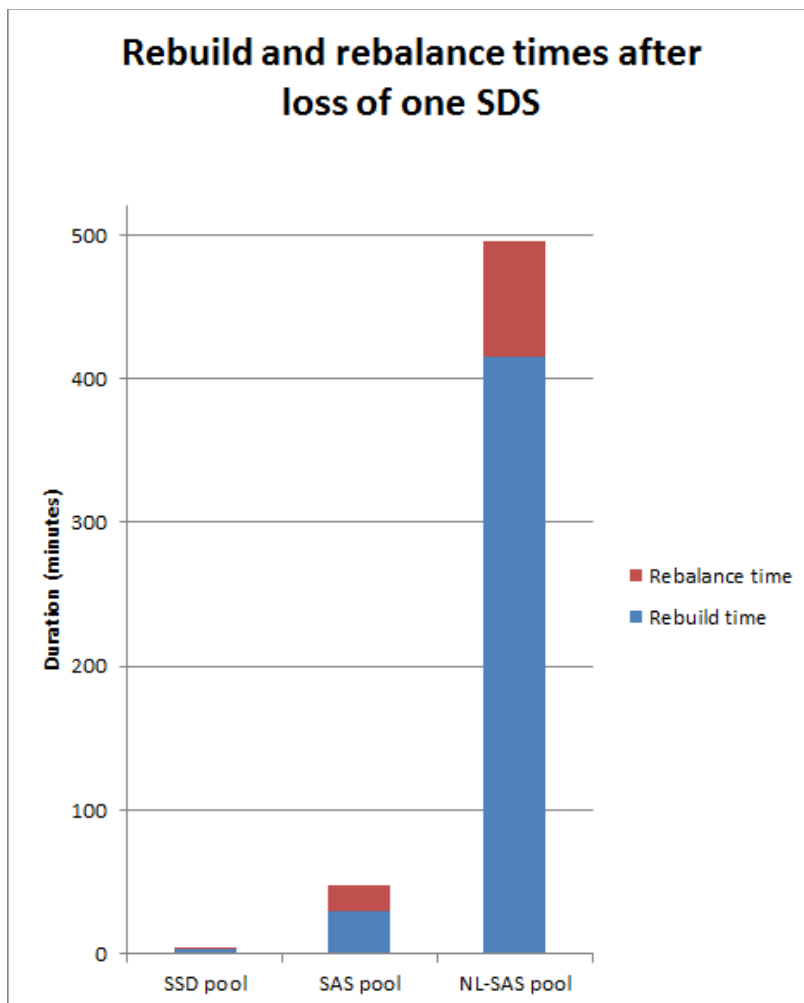


#### Rebuild/rebalancing capacities and durations following loss of a Tornado++ SSD/SAS SDS

The relative rebuild and subsequent rebalance times (by pool type) observed in testing when a fully utilized SDS fails are shown in the bar chart below. Note that

the NL-SAS pool rebuild time can be very high, as these servers have a very large capacity, and very limited IOPS capability.

The SSD and SAS pools rebuild times are acceptably fast.



**Relative rebuild / rebalance times by pool for loss of a single SDS**

## 12 Appendix B - ScaleIO on Tornado++ : hardware MTBF

The following table summarizes the estimated Mean-Time-Between-Failure (MTBF) for ScaleIO pools running on the Tornado++ configuration, based on an analysis of hardware failure scenarios. Note, however, that human error is a more likely source of data unavailability than hardware failure (e.g. rebooting a server at the wrong time). Software-related will also play a part in the overall system MTBF.

Four different double-hardware-failure scenarios have been considered, using a ScaleIO reliability model developed by EMC, in order to estimate the MTBF for ScaleIO pools running on the Tornado++ platform.

| Failure scenario           | SSD pool MTBF | SAS pool MTBF | NL-SAS pool MTBF |
|----------------------------|---------------|---------------|------------------|
| Disk-then-disk failure     | 212877 years  | 10736 years   | 8011 years       |
| Disk-then-server failure   | 202403 years  | 71347 years   | 33975 years      |
| Server-then-disk failure   | 67468 years   | 11416 years   | 4854 years       |
| Server-then-server failure | 202403 years  | 228311 years  | 58242 years      |

**MTBF for ScaleIO pools on Tornado++ by failure scenario**

Note how different pools are more or less vulnerable to different failure scenarios depending on the number of servers in the pool, number of disks per server, rebuild performance of underlying storage, etc.

The above MTBF estimates are based on the following input parameters:

- Mean-Time-To-Failure (MTTF) for disks:
  - o 800,000 hours
- Mean-Time-To-Failure (MTTF) for servers:
  - o 250,000 hours

Mean-Time-To-Repair (MTTR) for servers (from testing):

- o SSD pool: 3 mins (low as SSD drives are fast and 1 server is only 1/48<sup>th</sup> of the pool)
- o SAS pool: 25 mins
- o NL-SAS pool: 7 hours (high as NL-SAS drives are large and slow and 1 server is 1/8<sup>th</sup> of the pool)

- Mean-Time-To-Repair (MTTR) for disks (estimated):
  - o SSD pool: 1 min
  - o SAS pool: 4 mins
  - o NL-SAS pool: 1 hour

Note that a Markov failure model was also used as a comparison (sanity check) with the EMC model and this yielded the following figures, which despite being calculated with a completely different model, are broadly similar to the EMC model.

| Failure scenario                                             | SSD pool MTBF | SAS pool MTBF | NL-SAS pool MTBF |
|--------------------------------------------------------------|---------------|---------------|------------------|
| Markov Model<br>Disk-then-disk with<br>rack-level fault sets | 140837 years  | 7130 years    | 5283 years       |

**MTBF for ScaleIO pools on Tornado++: Alternate Markov model**

## 12.1 Comparison with traditional storage arrays

The above analysis is not applicable to traditional storage array that are not based on independent servers. The closest comparison we can make is for the disk-then-disk double failure scenario. This is deemed appropriate as this failure scenarios is one of the worst case double failures scenarios (following the slightly worse disk-then-server scenario).

Citi now configures its traditional storage arrays with RAID10 rather than the much more reliable RAID6, as the latter has an unacceptably high performance overhead. ScaleIO also uses RAID10, and hence a direct disk-then-disk comparison is possible.

Here, ScaleIO scores well compared with the traditional storage arrays as the MTTR is very fast following a disk failure due to ScaleIO's many-to-many recovery process. For example, for a 600GB drive, a Symmetrix may take up to 4 hours to reprotect following a disk failure (many-to-one copy) whereas ScaleIO will complete the process in a few minutes. Also, the ScaleIO SAS drives have been split into 3 separate pools, whereas in a Symmetrix all the devices on the same type will typically be placed into a single large thinly-provisioned pool.

For example, comparing the ScaleIO and Symmetrix solutions for the SAS (600GB drive) tier and the disk -then-disk scenario:

- ScaleIO – MTBF for 3 \* 320 600GB SAS disk drive RAID10 pools: 10736 years per pool\*
- VMAX – MTBF for 1 \* 960 600GB SAS disk drive RAID10 pool: 10995 years for pool\*\*

\* source: EMC's disk-then-disk ScaleIO model

\*\* source: <http://wintelguy.com/raidmttdl.pl> using 750,000 MTBF for disks, 100MB/sec throughput for drives, 4 hours to replace a failed drive and 80% (high) rebuild rate.

## 13 Appendix C – Networking Related Packages Required for Tornado++

The network configuration for the ScaleIO nodes involves two network connections on the front side (client network) and two connections on the back end (cluster private network). This is described in detail in section 9.6 above. In order to accommodate this, two packages were created. Details of the purpose and use of these packages is described in the following sections

### **13.1 Network carrier detection failure**

The carrier-mon package detects if a physical network adapter's carrier is down and if so will bring the corresponding network interface down. It is required to ensure that routing works correctly. Full details of the package and usage are listed below

**Package Name:** carrier-mon-1.0-A1.el6.noarch

**Pre-requisites:** RHEL6

**Script:** carrier-mon

**Original Creator:** Bob Mader/minor mods by Chris Howarth

**Packager:** Chris Howarth

Description: The carrier-mon package monitors the status of the carrier for a defined list of network adapters. If the carrier is detected as down then the logical adapter is brought down (ifdown). If the carrier comes back up, the logical adapter is brought back up (ifup)

#### **Usage:**

To configure carrier mon, a file with a list of adapters to be monitored needs to be created. For example:

```
# cat /etc/carrier-mon-nics.conf
eth0
eth1
eth2
eth3
```

To start monitoring an adapter:

```
initctl start carrier-mon NIC=<adapter>
```

For example:

```
# initctl start carrier-mon NIC=eth3
```

To stop monitoring an adapter:

```
# initctl stop carrier-mon NIC=eth3
```

To check the status of the monitor:

```
# initctl list | grep carrier-mon
start-carrier-mon stop/waiting
carrier-mon (eth3) start/running, process 2535
```

#### **Testing:**

To test, shutdown a network adapter and validate that the carrier-mon script starts it again:

For example:

```
# ip addr show eth3
5: eth3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 8800 qdisc mq state
UP qlen 1000
    link/ether 8c:dc:d4:aa:8d:b9 brd ff:ff:ff:ff:ff:ff
    inet 10.116.19.230/28 brd 10.116.19.239 scope global eth3
```

Shutdown the adapter and immediately verify it is briefly down:

```
# ifdown eth3; ip addr show eth3
5: eth3: <BROADCAST,MULTICAST> mtu 8800 qdisc mq state DOWN qlen
1000
    link/ether 8c:dc:d4:aa:8d:b9 brd ff:ff:ff:ff:ff:ff
```

Wait a couple of seconds:

```
# ip addr show eth3
5: eth3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 8800 qdisc mq state
UP qlen 1000
    link/ether 8c:dc:d4:aa:8d:b9 brd ff:ff:ff:ff:ff:ff
    inet 10.116.19.230/28 brd 10.116.19.239 scope global eth3
```

## 13.2 ScaleIO SDS Routing Configuration Package

Full details of the package are listed below. However at a high level, this package will configure routing entries for the four NICS adapters on each ScaleIO SDS node. In addition it will configure the default gateway for the nodes such that if the primary adapter on the client network fails, the default gateway switches to the secondary adapter. Note that the carrier-mon package described above is required to ensure that this failover takes place correctly.

**Package Name:** sdsroute-0.9-A0.el6.noarch

**Package Installation Prerequisites:**

- Citi standard RHEL 6.4 build
- Carrier-mon package installed

**Post-installation Usage:**

**Configure Routing**

Prerequisites:

- Four Ethernet interfaces connected to storage networks and configured with IP and MTU

- Network connectivity to ping geweb.nam.nsroot.net

#### Execution:

- As root run `# /opt/SDSroute/bin/sds_cfgroute`
  - o For verbose output messages, use the `-v` option
  - o To verify existing configuration files are accurate without making changes, use the `-F` option (can be run as non-root user)
- Reboot the host to activate changes

#### Changes made to configuration files:

- `/etc/sysconfig/network-scripts/route-<interface>` files are backed up and replaced
- `/etc/sysconfig/network-scripts/rule-<interface>` files are backed up and replaced
- `/etc/carrier-mon-nics.conf` file is backed up and replaced
- `/etc/sysctl.conf` file is backed up and appended to
- Copies of route, rule, and carrier-mon files are placed in `/opt/SDSroute/vault/`

### Verify Routing Configuration

#### Prerequisites:

- `sds_cfgroute` has been run and the host has been rebooted
- All four Ethernet interfaces are up and operational
- Network connectivity to ping geweb.nam.nsroot.net

#### Execution:

- As any user run `$ /opt/SDSroute/bin/sds_verifyroute`
  - o For verbose output messages, use the `-v` option

#### Verification checks made:

- Ethernet interface IP and MTU are verified to match defined storage networks
- `route-<interface>` and `rule-<interface>` files in `/etc/sysconfig/network-scripts` are verified
- `/etc/carrier-mon-nics.conf` is verified
- `/etc/sysctl.conf` is verified
- Verifies carrier-mon is running for each defined interface
- Verifies local gateway/router for each interface can be pinged
- For each storage network address block, verifies the first and last host IP will be routed over the correct interface
- Verifies a non-storage IP will be routed across default gateway on the first frontend interface

- Verifies ip route tables 1, 2, 5, and 7 have entries defined
- Verifies ip rules exist that direct to tables 1, 2, 5, and 7