ClusterStor™ NXD Administration Guide

(3.0.0.SU010)

S-2590

Contents

1 About the Cray NXD Administration Guide (S-2590)

The *Cray™ NXD Administration Guide* (3.0.0) S-2590 includes procedures to operate, manage, and troubleshoot systems that include the NXD feature. NXD uses flash acceleration software to improve performance for small block I/Os by caching them on SSD drives.

Table 1. Revision History

Publication Title	Date	Updates
NXD Administration Guide (3.0.0) S-2590	February 2018	Title changed and new Cray publication number added. Updated content for software release 3.0.0 SU-010. Some content has been reorganized.
Managing NXD on ClusterStor Systems (Seagate part number 1022038-02)	September 2017	Updated section on managing a GridRAID resource and added information about running Lustre filesystem integrity checks (e2fsck).
Managing NXD on ClusterStor Systems (Seagate part number 1022038-01)	June 2017	Initial Seagate publication.

Scope and Audience

This publication is written for system administrators tasked with operating, managing, and troubleshooting the NXD feature.

Typographic Conventions

`Monospace`	Indicates program code, reserved words, library functions, command-line prompts, screen output, file/path names, and other software constructs.
`Monospaced Bold`	Indicates commands that must be entered on a command line or in response to an interactive prompt.
`Oblique` or *`Italics`*	Indicates user-supplied values in commands or syntax definitions.
Proportional Bold	Indicates a **GUI Window**, **GUI element**, cascading menu (**Ctrl**→**Alt**→**Delete**), or key strokes (press **Enter**).
\ (backslash)	At the end of a command line, indicates the Linux® shell line continuation character (lines joined by a backslash are parsed as a single line).

Other Conventions

Sample commands and command output used throughout this publication are shown with a generic filesystem name of **cls12345**.

Trademarks

2 Cray NXD Feature

This publication describes how to operate, manage, and troubleshoot systems that are configured to use NXD.

NXD uses flash acceleration software to improve performance for small block I/Os by caching them on SSD drives.

Procedures in this section are written specifically for use by an admin user and do not require root (super user) access. We recommend that these procedures be followed as written and that the technician does not log in as root or perform the procedures as a root user. Adding the feature requires approximately one hour.

Overview

NXD implements write back caching on SSD drives for I/O on small blocks. The NXD functionality is characterized by the following:

- Cache is populated by writes that are less than or equal to the size of a user configured I/O bypass size. These writes are referred to as small block I/O operations.
- Writes larger than the configured bypass IO size are referred to as large block I/O operations and completely bypass the cache. However, if the large block I/O request overlaps data that is already cached, it is treated as a cached I/O. This is done to ensure data integrity.
- Reads on cached data are served from the cache.
- Small block reads on non-cached data do not populate the cache. They are served directly from the backend disks where the data resides.

NXD Theory of Operation

One of the core components of NXD is a kernel I/O filter driver that is implemented as a Linux device mapper (DM) target driver. This driver intercepts I/O and routes small blocks for caching. Large blocks bypass caching and are directly handed off to the underlying device (GridRAID).

Caching is implemented as a cache management library with well-defined APIs, deployed as a Linux kernel module.

NXD's I/O and caching capability operates at the Linux kernel block layer, which makes it transparent to the filesystem, database applications, and other applications.

NXD is hardware agnostic and can work with any block device.

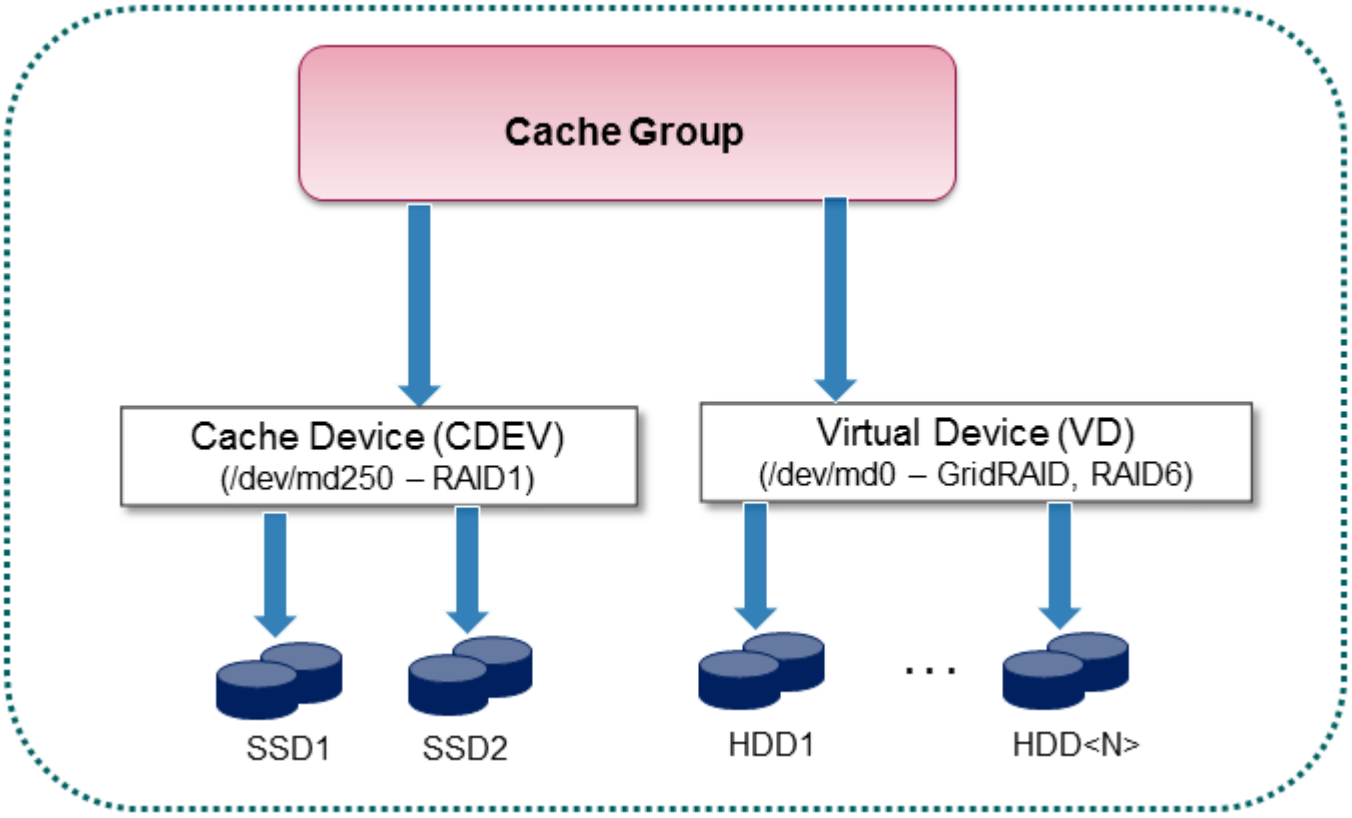
Cache Group

NXD introduces the concept of a cache group, which is a collection of:

- A single GridRAID data volume to be cached, called a virtual device or virtual drive (VD), usually `/dev/md`**x** (RAID6), where X=0, 2, 4... for even numbered OSS nodes and 1, 3, 5... for odd numbered OSS nodes.

- A single cache/caching device (CDEV) in which data is cached. The CDEV in a system usually shares the same set of SSDs that are used for WIB and JRNL. The NXD configuration uses larger capacity SSDs (typically 3.2 TB). The CDEV is configured as a single RAID1 md device (in the case of 2 SSDs) or RAID10 md device (in the case of 4, 6, 8, or 10 SSDs). For example, `/dev/md25`**`x`**, where X = 0, 2, 4… for even numbered OSS nodes and 1, 3, 5 … for odd numbered OSS nodes.

Figure 1. NXD Cache Group



Each instance of GridRAID shall be in a separate cache group (i.e., each OST will have its own cache group). Therefore, in an SSU+0 configuration an OSS node will show the following sample output (condensed format) from the `cscli nxd list` command.

```
[admin@cls12345n000 ~]$ cscli nxd list
--------------------------------------------------------------------------------
-----
Host         Cache       Caching Total    Cache      Cache    Cache       Bypass
             Group       State   Cache    Size       Block    Window      IO Size
                                 Size     In Use     Size     Size
--------------------------------------------------------------------------------
-----
cls12345n004  nxd_cache_0 enabled 1.406 TB 699.875 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
cls12345n005  nxd_cache_1 enabled 1.406 TB 745.500 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
--------------------------------------------------------------------------------
-----
```

In the above output:

- The NXD Caching State is “enabled”, indicating that NXD caching is in effect.
- Cache Block Size, Cache Window Size, and Bypass IO Size are displayed in sectors (where the size of one sector is 512 bytes), as well as in kibibytes (in parentheses).

3 NXD Supported Features

The following NXD features have been implemented in this software release.

- Write back cache
- Small block random acceleration - 4 KiB to 64 KiB
- Optimized for large block sequential (128 KiB to 1 MiB and above)
- Increase HDD throughput
- Advanced resource (RAM, SSD) management
- Tuning of Cache Bypass Size
- Automatic flushing
- Cache metadata and metadata logging
- Device I/O error handling
- Dirty data ‘flush’ logic in cache, with 1 MB aligned writes to GridRAID - increases performance
- Cache sub-states to dynamically switch between caching and bypass modes of handling I/O
- Read persistence (turned ON by default)
- GridRAID and HA support
- Command Line Interface (CLI) - Management
- Performance monitor - cache hits, cache performance etc.
- Performance data in JSON format
- Enable and disable caching dynamically
- Telemetry support
- Support for multi-node deployment (OST, NSD)
- Secure Linux support - (supports Secure Data appliances)

3.1 NXD Hardware and Operating System Requirements

Hardware Requirements

- ClusterStor L300N systems running software release 3.0.0
- SSUs with 64GB RAM or above. Typically, NXD requires:
  - 64GB RAM for SSU+0 configurations with 2 SSDs (3.2TB each) RAID1 cache device
  - 128GB RAM for SSU+1, SSU+2, and SSU+3 configurations

Operating System Requirements

Linux CentOS 7.2 and above

4 Install and Enable NXD

Prerequisites

For the NXD feature to be supported and installed, the YAML file used to install the system during manufacturing must have the following options set to yes:

`NXDSupport: <yes|no> : yes`

Controls the partitioning of SSDs to support NXD:

- A value of **`yes`** indicates SSDs will be properly partitioned to support creation of NXD cache devices.
- A value of **`no`** indicates that the system will not be configured to use/support NXD.

`NXDEnable: <yes|no> : yes`

Controls the enabling of NXD for the cluster:

- A value of **`yes`** indicates that NXD will be installed.
- A value of **`no`** indicates that the SSDs are partitioned for NXD, but the NXD feature will not be installed.

When the `NXDEnable` option is set to **`yes`**, the `NXDSupport` option must also be set.

5 SSD Partitioning for NXD

The default system configuration for Lustre storage products is currently SSU+0, which has 2 NXD cache devices for 2 GridRAID arrays. The cache devices are created from SSDs, with the default configuration consisting of 2 SSDs. Each SSD is equally divided into 2 NXD partitions, along with WIB and JRNL partitions. 2 NXD partitions are taken from each SSD to create 2 RAID1 md devices, 1 for each GridRAID device (i.e., OST) in a single SSU.

Current system configurations that use Expanded Storage Units (ESUs) do not contain additional SSDs to cache additional GridRAID arrays. Therefore, additional partitions are created when the system is manufactured so that the available SSD space is distributed evenly between the NXD cache devices. For an SSU+1 configuration, 4 NXD partitions are taken from each SSD to create 4 RAID1 md devices, 1 for each GridRAID device in an SSU. In this instance, the NXD cache size is decreased compared to an SSU+0 configuration.

Similarly, for SSU+2 and SSU+3 configurations, 6 and 8 NXD partitions, respectively, are taken from each SSD to create 6 and 8 RAID1 md devices, 1 for each GridRAID device in an SSU. The NXD cache size is further decreased compared to SSU+0 and SSU+1 configurations.

Customers may choose to increase the NXD cache size by adding more SSDs to the configuration. The current software release supports 2, 4, 6, 8, or 10 SSDs per SSU, and a single NXD cache device size should be no larger than 3.2 TB. When more than 2 SSDs are installed in a single SSU, RAID10 md devices are created as NXD cache devices, 1 for each GridRAID device in a single SSU. The size of the RAID10 md cache device must be no larger than 3.2 TB.

6 Add and Remove the NXD Feature

This section describes how to add or remove the NXD feature to or from the data path.

Add the NXD Feature After It Has Been Removed

This procedure adds NXD into the data path and is equivalent to converting a non-NXD system to a system on which NXD may be enabled if desired. The procedure ensures that the filesystem and caches are started properly, to avoid data loss.

This process is valid only if the system was properly installed with `NXDSupport: yes` in the YAML file.

1. Stop the Lustre filesystem:

   ```
   [MGMT0]$ cscli unmount
   ```

2. Start the NXD service:

   ```
   [MGMT0]$ cscli nxd service start
   ```

   This command starts the NXD service on all storage nodes.

3. Start the Lustre filesystem:

   ```
   [MGMT0]$ cscli mount
   ```

   This command brings up all the RAID resources from all nodes and mounts the filesystem.

   NXD caching may now be enabled on the system if desired.

Remove the NXD Feature After It Has Been Added

This procedure completely removes the NXD feature from the data path and ensures that the filesystem and caches are shut down properly, to avoid data loss.

1. Disable NXD caching before removing NXD from the data path, and then wait for the cached dirty data to be flushed to the main array:

   ```
   [MGMT0]$ cscsli nxd disable
   ```

   This command initiates a process to flush the NXD cache to the main array. While the cached dirty data is being flushed, running the `cscli nxd list` command will show the value `disabling` in the **Caching State** field. This value continues to show until NXD has completed the flush process.

   The NXD flush process proceeds in the background. After all of the data to be flushed from the cache has been written out to the drives on the main array, the flush process is deemed complete and NXD disables caching on all nodes.

Once NXD caching is disabled, output from the `cscli nxd list` command, should show the value `disabled` in the **Caching State** field. To confirm the completion of this step, run the `cscli nxd list` command repeatedly to poll for the value `disabled` in the **Caching State** field.

Note that if the cache is completely full, it may take more than 40 minutes to flush all cached data out to drives, depending on the following factors:

- The size of the cache device
- The amount of cached data that must be flushed
- The state of the underlying main array (GridRAID). It may take longer to write all cached data from SSD to a degraded array (GridRAID) or when the GridRAID array is already overloaded to service large bypassed I/Os.

NXD does a fast flush as part of the `cscsli nxd disable` command, at a flush rate of 500 MiB/Sec or above.

Sometimes the caching state will remain in the “disabling” state for a considerable period of time if client I/O has not stopped. This may occur when data in new I/O operations overlaps with data that is already in the cache. As a result, it will take longer to flush all of the cached data. Run `cscli nxd list -a` to see more details about the cache state, Dirty CWs, Cache Blocks Flushed, etc.

2. Stop the Lustre filesystem:

```
[MGMT0]$ cscli unmount
```

3. Stop the NXD service:

```
[MGMT0]$ cscli nxd service stop
```

This command stops the NXD service on all storage nodes.

4. Start the Lustre filesystem:

```
[MGMT0]$ cscli mount
```

7 CSCLI Commands to Manage NXD

The CSCLI `nxd` command and its subcommands are available to control and manage NXD on all OSS nodes while NXD is active and running on them. For the most up to date information about NXD commands and options, please see the latest revision of *CSCLI Reference Guide*, which is available at *http://pubs.cray.com*.

In general, use the CSCLI commands to:

- Verify that NXD is running properly on the OSS nodes
- Manage and monitor NXD

The NXD commands are available only if the cluster is configured for NXD and it has been enabled.

All NXD commands operate on all of the cluster's OSS nodes simultaneously. Specifically, it is not possible to enable NXD cache on one OSS node and disable NXD cache on another OSS node. Similarly, it is not possible to start NXD service on one OSS node and stop it on another OSS node. This rule also applies when modifying the bypass I/O size.

7.1 `nxd` Command

The `nxd` command and subcommands are used to manage the NXD feature, which improves performance for small block I/Os by caching them on SSD drives.

The full set of NXD subcommands is available only if the cluster is configured for NXD and it has been enabled. If NXD is configured but not enabled, only the `cscli nxd service` subcommand is available.

Synopsis

```
$ cscli nxd [-h] {service,enable,disable,list,modify} ...
```

where:

Positional Arguments	Description
`service`	Start, stop, and check the status of the NXD service
`enable`	Enable caching on all OSS nodes.
`disable`	Disable caching on all OSS nodes.
`list`	Displays basic NXD configuration.
`modify`	Modify configurations for NXD.

Optional Arguments	Description
`-h |--help`	Displays the help message and exits.

Usage

```
$ cscli nxd [-h] service {status | start | stop}
$ cscli nxd [-h] {enable | disable}
$ cscli nxd [-h] list [-a | --advance] [{-cg | --cache_group} <cg_name>]
$ cscli nxd [-h] modify --bypass_size BYPASS_SIZE
```

7.2 Start and Stop the NXD Service

Use the `cscli nxd service` command to start and stop the NXD service on all nodes.

By default, NXD runs on NXD-enabled sysytems. If it is desired to run a given workload in non-NXD mode, remove NXD from the data path by stopping the NXD service on all OSS nodes

The following is sample output for stopping NXD service on all (OSS) nodes:

```
[admin@cls12345n000 ~]$ cscli nxd service stop
nxd: NXD Service update status: Successful.
Details: Service stopped successfully.
```

Attempting to stop the NXD service without first properly unmounting the filesystem will give the error:

```
[admin@cls12345n000 ~]$ cscli nxd service stop
nxd: Error: Lustre targets are still mounted. Please unmount
Lustre targets before changing NytroXD Service.
To do this please use UI or 'cscli unmount' command.
```

The following is sample output for starting the NXD service on all (OSS) nodes:

```
[admin@cls12345n000 ~]$ cscli nxd service start
service: Nxd Service update: Successful.
Details: Service started successfully.
```

All NXD commands operate on all of the cluster's OSS nodes simultaneously. Specifically, it is not possible to enable NXD cache on one OSS node and disable NXD cache on another OSS node. Similarly, it is not possible to start NXD service on one OSS node and stop it on another OSS node. This rule also applies when modifying the bypass I/O size.

7.2.1 `nxd service` Subcommand

The `nxd service` command is a subcommand of the `nxd` command, and is used to start and stop the NXD service across all OSS nodes, and to display the status of the NXD service.

Synopsis

```
$ cscli nxd service [-h] {status,start,stop} ...
```

where:

Positional Arguments	Description
`status`	Display the status of the NXD service across all (OSS) nodes.
`start`	Start the NXD service across all (OSS) nodes.
`stop`	Stop the NXD service across all (OSS) nodes.

Optional Arguments	Description
`-h |--help`	Displays the help message and exits.

7.2.1.1 `nxd service start` Subcommand

The `nxd service start` command is a second level subcommand of the `nxd` command, and is used to start the NXD service across all OSS nodes.

Synopsis

```
$ cscli nxd service start [-h]
```

where:

Optional Arguments	Description
`-h |--help`	Displays the help message and exits.

7.2.1.2 `nxd service stop` Subcommand

The `nxd service stop` command is a second level subcommand of the `nxd` command, and is used to stop the NXD service across all OSS nodes.

Synopsis

```
$ cscli nxd service stop [-h]
```

where:

Optional Arguments	Description
`-h |--help`	Displays the help message and exits.

7.3 Determine the Current Status of NXD Service in the Cluster

Use the `cscli nxd service status` command to display whether the NXD service is running or stopped on all OSS nodes.

Output for a system where the NXD Service is started:

```
[admin@cls12345n000 ~]$ cscli nxd service status
nxd: NytroXD Service.
Status: Running on All nodes.
```

Output for a system where the NXD Service is stopped:

```
[admin@cls12345n000 ~]$ cscli nxd service status
nxd: NytroXD Service.
Status: Stopped on All nodes.
```

Alternative Node-specific Command

Alternatively, the node-specific command `service nytroxd status` may be run remotely on all the OSS nodes to obtain the current status of the NXD service on the nodes, as shown in the example below:

Output for a system where the NXD Service is started:

```
[admin@cls12345n000 ~]$pdsh -w cls12345n[004-005] service nytroxd status | dshbak -c
cls12345n004: Redirecting to /bin/systemctl status  nytroxd.service
cls12345n005: Redirecting to /bin/systemctl status  nytroxd.service
----------------
cls12345n004
----------------
● nytroxd.service - loads nytroxd kernel modules if not loaded and then starts the nytroxd dev monitor
   Loaded: loaded (/etc/sba/nytroxd; static; vendor preset: disabled)
   Active: active (running) since Mon 2018-01-08 10:52:04 CST; 4 days ago
  Process: 28164 ExecStop=/etc/sba/nytroxd stop (code=exited, status=0/SUCCESS)
  Process: 28188 ExecStart=/etc/sba/nytroxd start (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/nytroxd.service
           └─28233 /opt/seagate/nytroxd/nytroxd-dev-monitor
----------------
cls12345n005
----------------
● nytroxd.service - loads nytroxd kernel modules if not loaded and then starts the nytroxd dev monitor
   Loaded: loaded (/etc/sba/nytroxd; static; vendor preset: disabled)
   Active: active (running) since Mon 2018-01-08 10:52:04 CST; 4 days ago
  Process: 3199 ExecStop=/etc/sba/nytroxd stop (code=exited, status=0/SUCCESS)
  Process: 3246 ExecStart=/etc/sba/nytroxd start (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/nytroxd.service
           └─3296 /opt/seagate/nytroxd/nytroxd-dev-monitor
```

Output for a system where the NXD service is stopped:

```
[admin@cls12345n000 ~]$pdsh -w cls12345n[004-005] service nytroxd status | dshbak -c
cls12345n004: Redirecting to /bin/systemctl status nytroxd.service
cls12345n005: Redirecting to /bin/systemctl status nytroxd.service
pdsh@cls12345n000: cls12345n004: ssh exited with exit code 3
pdsh@cls12345n000: cls12345n005: ssh exited with exit code 3
----------------
cls12345n[004-005]
----------------
● nytroxd.service - loads nytroxd kernel modules if not loaded and then starts the nytroxd dev monitor
Loaded: loaded (/etc/sba/nytroxd; static; vendor preset: disabled)
Active: inactive (dead) since Wed 2018-01-31 12:44:52 CST; 16s ago
```

7.3.1 `nxd service status` Subcommand

The `nxd service status` command is a second level subcommand of the `nxd` command, and is used to display the status of the NXD service across all OSS nodes.

Synopsis

```
$ cscli nxd service status [-h]
```

where:

Optional Arguments	Description
`-h |--help`	Displays the help message and exits.

7.4 Enable and Disable NXD Caching

When caching is disabled, NXD remains active in the system. The NXD kernel drivers and user processes continue to run on the system, and the system memory allocated for caching remains with NXD. The only difference between enabled and disabled is:

- When caching is enabled using the `cscli nxd enable` command, NXD caches small blocks in SSD, which are less than or equal to the configured bypass size.
- When caching is disabled using the `cscli nxd disable` command, small blocks will not be cached. They "bypass" the caching layer in NXD.

Use the `cscli nxd list` command to verify whether the caching state has changed to **enabled**.

Note that there is no output from the `cscli nxd enable` and `cscli nxd disable` commands. Use the `cscli nxd list` command to verify whether the caching state has changed to **disabled**.

7.4.1 `nxd enable` Subcommand

The `nxd enable` command is a subcommand of the nxd command. Use the command to enable caching globally on all SSUs.

Synopsis

```
$ cscli nxd enable [-h]
```

where:

Optional Arguments	Description
`-h |--help`	Displays the help message and exits.

7.4.2 `nxd disable` Subcommand

The `nxd disable` command is a subcommand of the nxd command. Use the command to disable caching globally on all SSUs.

Synopsis

```
$ cscli nxd disable [-h]
```

where:

Optional Arguments	Description
`-h |--help`	Displays the help message and exits.

7.5 Display NXD Details of OSS Nodes

The `cscli nxd list` command shows the related details of the OSS nodes on which NXD is installed, and indicates whether NXD caching is enabled:

```
[admin@cls12345n000 ~]$ cscli nxd list
------------------------------------------------------------------------------------------
-----
 Host       Cache       Caching   Total   Cache      Cache     Cache      Bypass
            Group       State     Cache   Size       Block     Window     IO Size
                                  Size    In Use     Size      Size
------------------------------------------------------------------------------------------
-----
cls12345n004 nxd_cache_0 enabled  1.406 TB 699.875 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
cls12345n005 nxd_cache_1 enabled  1.406 TB 745.500 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
------------------------------------------------------------------------------------------
-----
```

In the above output:

- The NXD Caching State is “enabled”, indicating that NXD caching is in effect.
- Cache Block Size, Cache Window Size, and Bypass IO Size are displayed in sectors (where the size of one sector is 512 bytes), as well as in kibibytes (in parentheses).

Use the advanced option on the `cscli nxd list` command to display detailed NXD configuration information and statistics.

```
 [admin@cls12345n000 ~]$ cscli nxd list -a
------------------------------------------------------------------------------------------
--------------------
 Host         Cache Group  Reads     Writes    Bypass   Bypass  Cache
Cache     Dirty  Cache Blocks
                                               reads    writes  hits
Misses    CWs    Flushed
------------------------------------------------------------------------------------------
--------------------
 cls12345n004 nxd_cache_0  522846454 341655881 39879669  0      776530687
48091979   0     935564050
 cls12345n005 nxd_cache_1  522888813 341581093 34674799  0      785675820
44119287   0     941760819
------------------------------------------------------------------------------------------
--------------------
```

To display additional details about a specific cache group, run the `cscli nxd list` command with the cache group filter option. This option is useful when focusing only on odd or even OSTs in a multi-SSU system. The following sample output shows an example of obtaining I/O performance statistics on a specific cache group (`nxd_cache_1` in this case):

```
[admin@cls12345n000 ~]$ cscli nxd list -a -cg nxd_cache_1
------------------------------------------------------------------------------------------
---
Host          Cache Group    Reads   Writes Bypass Bypass Cache Cache  Dirty Cache
                                            reads  writes Hits  Misses CWs   Blocks

Flushed
------------------------------------------------------------------------------------------
----
cls12345n005   nxd_cache_1   293001  149     293001  0    1      148    0    149
------------------------------------------------------------------------------------------
---
```

7.5.1 `nxd list` Subcommand

The `nxd list` command is a subcommand of the `nxd` command. Use the command to display the basic NXD configuration of all the OSS nodes.

Synopsis

```
$ cscli nxd list [-h] [-a] [-cg CACHE_GROUP]
```

where:

Optional Arguments	Description
`-a | --advance`	Displays detailed NXD configuration and statistics.
`-cg CACHE_GROUP` `| --cache_group CACHE_GROUP`	Displays Configurations of given Cache group.
`-h |--help`	Displays the help message and exits.

7.6 Modify “Small Block” Size in NXD

The default size of small blocks cached by the NXD feature is 32 KiB. This means:

- All I/Os smaller than or equal to 32 KiB are considered “small blocks” and will be cached.
- All I/Os larger than 32 KiB will bypass the caching layer in NXD.

The small block size, however, is a tunable parameter that may be specified using the `cscli nxd modify` subcommand with the `--bypass_size` option, by supplying a value in units of 512-Byte sectors.

Please note that NXD caching must be disabled and the filesystem stopped before modifying the "Small Block" size. For a change in the small block size to take effect, the NXD service must be restarted across all nodes in the cluster. This service restart requires no user intervention. When small block size is changed using the `cscli nxd modify` subcommand, the subcommand restarts the NXD service automatically.

Following is sample output when changing the `BYPASS-SIZE` from the default value of 32 KiB to 64 KiB:

```
[admin@cls12345n000 ~]$ cscli nxd modify --bypass_size 128
xd: Please wait while updating parameters...
nxd: IEC: 014008004: NXD Parameter changed:

{"from_io_bypass_size": 64, "to_io_bypass_size": 128}
nxd: NXD Parameters Update: Success.
```

The `BYPASS-SIZE` of 128 that is passed to the `cscli nxd modify` subcommand in the above example is the number of 512-Byte sectors in 64 KiB.

If the `cscli nxd modify` subcommand determines that the filesystem is still mounted the command will provide a warning that the filesystem must be unmounted first, and then the command exits. For example, on a Lustre filesystem mounted on the NXD device, the following error appears:

```
[admin@cls12345n000 ~]$ cscli nxd modify --bypass_size 128
nxd: Error: Lustre targets are still mounted. Please unmount Lustre
targets before changing NytroXD Parameters.
To do this please use UI or 'cscli unmount' command.
```

In the above case, use the following steps to check if the filesystem is mounted and, if necessary, unmount the filesystem before running the `cscli nxd modify` subcommand:

1. To check if the Lustre filesystem is mounted, run the `cscli show_nodes` command.
2. Verify that the NXD cached state is DISABLED by running the `cscli nxd list` command.

   When the cache state is DISABLED there is no outstanding dirty data in the cache device, and it is safe to unmount Lustre.
3. To unmount the Lustre filesystem, run the `cscli unmount` command.

7.6.1 `nxd modify` Subcommand

The `nxd modify` command is a subcommand of the `nxd` command. Use the command to set the tunable parameter `bypass_size`. The value takes effect when the NXD service restarts.

The filesystem must be taken offline before modifying the tunable parameters.

Synopsis

```
$ cscli nxd modify [-h] [--bypass_size BYPASS_SIZE]
```

where:

Optional Arguments	Description
`--bypass_size BYPASS_SIZE`	Updates default configuration of I/O bypass size. The values for `bypass_size` must be within 32 (16K) to 2048 (1M) and power of 2 (32,64,128,256,....).
`-h |--help`	Displays the help message and exits.

8 Start Lustre File System on NXD-Enabled Systems

NXD is installed only on the OSS nodes of Lustre storage systems. Lustre is started by running the `cscli mount` command from the primary management node (same as non-NXD systems). This command brings up the underlying Object Storage Targets, or OSTs (RAID devices), on each node and then mounts the filesystem. There is a difference between the underlying devices for OSTs on non-NXD versus NXD-enabled systems:

- non-NXD systems use standard GridRAID devices (`/dev/mdX`)
- NXD-enabled systems use device-mapped GridRAID devices (`/dev/dm-X`)

Running the `cscli show_nodes` command will display the current status of the OSS nodes and their OSTs (GridRAID or device-mapped GridRAID), such as:

- Power state (On/Off)
- Service state (Started/Stopped)
- Number of targets

Note that NXD must be enabled again after the filesystem is mounted if it is desired to continue caching small block I/O operations.

To start Lustre and enable NXD:

1. Start the Lustre filesystem.

```
[MGMT0]$ cscli mount
```

2. Use the `cscli show_nodes` command to monitor the progress of bringing up services on the Lustre nodes. Note that it may take a few seconds after issuing the `cscli mount` command for the `show_nodes` command output to reflect the correct/updated status.
3. Continue running the `cscli show_nodes` command until the output shows that all services have started, as shown in the sample output below.

```
[admin@cls12345n000 ~]$ cscli show_nodes
--------------------------------------------------------------------------------
---------
Hostname      Role       Power State    Service State Targets  HA Partner    HA
Resources
--------------------------------------------------------------------------------
---------
cls12345n000  MGMT       On             N/a           0 / 0    cls12345n001
None
cls12345n001   (MGMT)    On             N/a           0 / 0    cls12345n000
None
cls12345n002  MDS,MGS    On             Started       1 / 1    cls12345n003
Local
cls12345n003  MDS,(MGS)  On             Started       1 / 1    cls12345n002
Local
cls12345n004  OSS        On             Started       1 / 1    cls12345n005
Local
```

```
cls12345n005  OSS         On              Started       1 / 1    cls12345n004
Local
-------------------------------------------------------------------------------
---------
```

Because of drive or hardware issues, one or more targets may occasionally show a **Service State** of `Stopped`, often when a failure occurs trying to bring up the underlying RAID arrays for those nodes. It may be necessary to triage the issue by manually inspecting the affected nodes.

4. Once the filesystem is mounted and all services started, enable the caching of small block I/O operations.

```
[admin@cls12345n000 ~]$ cscli nxd enable
```

9 Stop Lustre Filesystem on NXD-Enabled Systems

This procedure describes how to stop the Lustre filesystem on an NXD-enabled system. Note that NXD caching must be disabled before stopping the filesystem. The procedure assumes that NXD is already enabled.

1. Stop all client I/O to the system.
2. Stop NXD caching on the system.

```
[admin@cls12345n000 ~]$ cscli nxd disable
```

3. Monitor the NXD caching state by running the `cscli nxd list` command repeatedly and checking the output.

```
[admin@cls12345n000 ~]$ cscli nxd list
```

   Wait until the value displayed in the **Caching State** field is `disabled` for all OSS nodes and OSTs, before proceeding to the next step.

   If there is a valid reason not to wait for the caching state to become disabled on all OSS nodes and OSTs, then use the `--nxdforce` option on the `cscli unmount` command in the next step. Be aware that this option could result in data corruption

4. Stop Lustre.

```
[admin@cls12345n000 ~]$ cscli unmount
```

   The command will check first to make sure that NXD caching has been disabled on all OSS nodes and OSTs. If not, the command will not unmount the filesystem. To bypass the check and force the filesystem to unmount, add the `--nxdforce` option.

```
[admin@cls12345n000 ~]$ cscli unmount --nxdforce
```

CAUTION: Using the `--nxdforce` option could result in data corruption.

Warnings

- Do not run a file system integrity check on GridRAID arrays after unmounting Lustre using the `--nxdforce` option. This is because data in the NXD cache will not be flushed completely.
- Do not stop the NXD service after unmounting Lustre using the `--nxdforce` option.
- In the event of faulty or degraded OSTs occurring, the recovery process should be performed in a controlled environment after consulting with appropriate support and engineering personnel.

10 NXD Cache State on Lustre Filesystem

When the Lustre filesystem is mounted, the NXD cache state may be enabled or disabled on **all** of the Object Storage Targets (OSTs). There is no option to change the NXD cache state on selected OSTs only.

Use the `cscli nxd list` command to determine if NXD caching on OSTs is enabled or disabled. In the following example, the caching state is enabled on the targets:

```
[root@cls12345n00 ~]$ cscli nxd list
------------------------------------------------------------------------------------------
-----
Host           Cache       Caching Total    Cache      Cache    Cache
Bypass
               Group       State   Cache    Size       Block    Window     IO Size
                                   Size     In Use     Size     Size
------------------------------------------------------------------------------------------
-----
cls12345n004   nxd_cache_0 enabled 1.406 TB 699.875 MB 8(4 KiB) 128(64 KiB) 2048(1
MiB)
cls12345n005   nxd_cache_1 enabled 1.406 TB 745.500 MB 8(4 KiB) 128(64 KiB) 2048(1
MiB)
------------------------------------------------------------------------------------------
-----
```

As needed, the NXD caching state may be changed using the `cscli nxd [enable|disable]` commands, however, the Lustre filesystem must be mounted. Note also that NXD caching may be enabled or disabled even if:

- An OSS node is down
- One or more OSTs of an OSS node are down

Note that dirty data may remain in the NXD cache if an OST goes down while NXD caching is enabled.

Considerations When Disabling NXD Caching

After running the `cscli nxd disable` command, it is possible for the output of the `cscli nxd list` command to continue showing the caching state for any OST as `disabling` for an extended period. This is because of the length of time needed by background processes to flush outstanding dirty data from the NXD cache device (SSD) to GridRAID.

On a healthy system it typically takes less than 40 minutes for 1.4TB of a cache device (90% of the dirty data) to flush completely to GridRAID. If the process takes longer, we recommend monitoring the flushing progress for the OST by using the `cscli nxd list -a` command repeatedly until the process is complete. The `-a` option will display detailed NXD configuration information and statistics in the **DirtyCWs** and **Cache Blocks Flushed** fields.

The following table sumarizes several common scenarios that may be observed when monitoring flushing progress:

Value in Dirty CWs Field is...	Value in Cache Blocks Flushed Field is...	Possible Issues
Not changing	Not changing	NXD cache device is up and running fine, but the NXD virtual drive (GridRAID) is faulty or down. Run `cat /proc/mdstat` to check GridRAID state on the affected OSS node.
Decreasing	Increasing	Wait until the caching state on the affected OSTs is completely disabled. The flushing speed might be reduced for several reasons, such as 1) GridRAID is busy serving large bypassed I/Os, 2) RAID check is running in the background, 3) GridRAID array is degraded due to 1 or 2 rotational drive failures.
Increasing	Increasing	There may be continuous large overlapping I/O coming into the affected OSTs. We recommend stopping client I/O until flushing is complete. Any new I/Os which are not overlapping in nature will bypass the NXD cache, but for any overlapping I/Os the only solution is to stop client I/Os to allow the NXD disable process to complete.

11 Manage NXD High Availability (HA)

The system's high availability (HA) functionality manages the NXD cache device as part of the GridRAID resource. For every GridRAID volume, a corresponding NXD cache device (RAID 1 MDRAID device constructed from two SSDs) is created and mapped. An HA utility handles the interaction with NXD for the HA resources, through which the NXD operation is controlled.

NXD HA is managed using the `cscli failover` and `cscli failback` commands.

Before a failover is initiated, the state of the NXD cluster is as follows:

```
[admin@cls12345n000 ~]$ cscli nxd list
------------------------------------------------------------------------------
-----
Host         Cache       Caching Total    Cache      Cache     Cache
Bypass
             Group       State   Cache    Size       Block     Window     IO Size
                                 Size     In Use     Size      Size
------------------------------------------------------------------------------
-----
cls12345n004 nxd_cache_0 enabled 1.406 TB 699.875 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
cls12345n005 nxd_cache_1 enabled 1.406 TB 745.500 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
------------------------------------------------------------------------------
-----

[admin@cls12345n000 ~]$ cscli show_nodes
------------------------------------------------------------------------------
-----
Hostname      Role       Power State    Service State Targets  HA Partner    HA
Resources
------------------------------------------------------------------------------
-----
cls12345n000  MGMT       On             N/a           0 / 0    cls12345n001   None
cls12345n001   (MGMT)    On             N/a           0 / 0    cls12345n000   None
cls12345n002  MDS,MGS    On             Started       1 / 1    cls12345n003   Local
cls12345n003  MDS,(MGS)  On             Started       1 / 1    cls12345n002   Local
cls12345n004  OSS        On             Started       1 / 1    cls12345n005   Local
cls12345n005  OSS        On             Started       1 / 1    cls12345n004   Local
------------------------------------------------------------------------------
-----
```

Following is sample output after a failover is initiated for the OSS host cls12345n004. It is expected to fail over to its OSS partner cls12345n005.

```
[admin@cls12345n000 ~]$ cscli failover -n cls12345n004
failover: Command completed successfully, process initiated.
failover: Use "cscli show_nodes" to see failover status.
```

Note that a failover or failback process takes a few minutes to take effect on an active filesystem. During failover or failback, the cache state of each OST is preserved. If there are cached dirty data present in any OST while flushing is in progress, the flushing will resume after the failover or failback, from the same point it was at when the failover or failback operation occurred.

Following is the corresponding output after the failover has completed successfully and cache group `nxd_cache_0` for OST0 (cls12345n004) has failed over to cls12345n005:

```
[admin@cls12345n000 ~]$ cscli show_nodes
-------------------------------------------------------------------------------
Hostname      Role     Power State Service State Targets HA Partner   HA Resources
-------------------------------------------------------------------------------
cls12345n000  MGMT      On          N/a           0 / 0   cls12345n001   None
cls12345n001  (MGMT)    On          N/a           0 / 0   cls12345n000   None
cls12345n002  MDS,MGS   On          Started       1 / 1   cls12345n003   Local
cls12345n003  MDS,(MGS) On          Started       1 / 1   cls12345n002   Local
cls12345n004  (OSS)     On          Stopped       0 / 1   cls12345n005   None
cls12345n005  OSS       On          Started       2 / 1   cls12345n004   All
-------------------------------------------------------------------------------

[admin@cls12345n000 ~]$ cscli nxd list
-------------------------------------------------------------------------------
-----
Host          Cache       Caching  Total     Cache      Cache    Cache       Bypass
              Group       State    Cache     Size       Block    Window      IO
Size
                                   Size      In Use     Size     Size
-------------------------------------------------------------------------------
-----
cls12345n005  nxd_cache_0 enabled  1.406 TB  699.875 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
cls12345n005  nxd_cache_1 enabled  1.406 TB  745.500 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
-------------------------------------------------------------------------------
-----
```

Following is the sample output after a failback is initiated, following a failover. In this example of a failback initiated on cls12345n04, it should restore all the services on cls12345n04 from its OSS partner cls12345n05.

```
[admin@cls12345n000 ~]$ cscli failback -n cls12345n004
failback: Command completed successfully, process initiated.
failback: Use "cscli show_nodes" to see failback status.
```

Finally, the following is the corresponding output after the failback has successfully completed:

```
[admin@cls12345n000 ~]$ cscli show_nodes
-------------------------------------------------------------------------------
-
Hostname     Role      Power State Service State  Targets HA Partner    HA
Resources
-------------------------------------------------------------------------------
-
cls12345n000 MGMT      On          N/a           0 / 0   cls12345n001  None
cls12345n001 (MGMT)    On          N/a           0 / 0   cls12345n000  None
cls12345n002 MDS,MGS   On          Started       1 / 1   cls12345n003  Local
cls12345n003 MDS,(MGS) On          Started       1 / 1   cls12345n002  Local
cls12345n004 OSS       On          Started       1 / 1   cls12345n005  Local
cls12345n005 OSS       On          Started       1 / 1   cls12345n004  Local
-------------------------------------------------------------------------------
-
```

```
[admin@cls12345n000 ~]$ cscli nxd list
-------------------------------------------------------------------------------------------
-----
Host          Cache       Caching  Total    Cache      Cache    Cache       Bypass
              Group       State    Cache    Size       Block    Window      IO
Size
                                   Size     In Use     Size     Size
-------------------------------------------------------------------------------------------
-----
cls12345n004  nxd_cache_0 enabled  1.406 TB 699.875 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
cls12345n005  nxd_cache_1 enabled  1.406 TB 745.500 MB 8(4 KiB) 128(64 KiB) 64(32
KiB)
-------------------------------------------------------------------------------------------
-----
```

12 NXD and Managing a GridRAID Resource

When stopping a GridRAID resource, the associated NXD cache device must be flushed completely to ensure that no "dirty data" remains in the cache. If dirty data is present in the associated cache device when the GridRAID resource is deactivated for any reason, NXD will start flushing the dirty data to the GridRAID resource when the resource is next activated. User data will be preserved in the cache device even if all of the dirty data have not yet landed to GridRAID.

Once all of the OST resource groups have been stopped, the RAID resources of the storage node can be activated, without starting the `fsys` resource, by using the following command on a given OSS node:

- To activate:

  ```
  mdraid-activate –d –m
  ```

  This command will bring all the RAID resources up for a given storage node (`–m` option for all my resources) without starting the `fsys` resource (`–d` option). For NXD-enabled systems, the mounted device is the device-mapped GridRAID volume (i.e., `/dev/dm-X`) and for normal systems without NXD enabled, the mounted device is a normal GridRAID volume (i.e., `/dev/mdY`).

- To deactivate:

  ```
  mdraid-deactivate –a
  ```

  This command will bring all the RAID resources down for a given storage node.

Run Lustre Filesystem Integrity Checks (`e2fsck`)

For NXD-enabled systems, filesystem integrity checks should be run on a device-mapped GridRAID volume (i.e., `/dev/dm-X`).

Example:

```
e2fsck –fvn /dev/dm-X
```

For systems without NXD enabled, filesystem integrity checks should be run on a normal GridRAID volume (i.e., `/dev/mdY`).

Example:

```
e2fsck –fvn /dev/mdY
```

Note that for non-NXD systems the mounted device is usually GridRAID (`/dev/md`**`x`**), where **`x`** is an even number for even-numbered nodes or an odd number for odd-numbered nodes. For NXD-enabled systems, the mounted device is device-mapped GridRAID.

Use the following command to determine the appropriate device mapper to use when there are multiple GridRAID arrays on a single OSS node:

```
OSS-Node> ls /sys/block/md0/holders/dm-0
OSS-Node> ls /sys/block/md2/holders/dm-1
```

13 NXD Logging

Any configuration change, error message, or other important event is reported by NXD. These items are logged in the system log, which is usually located in the primary management node directory `/mnt/mgmt/var/log/messages`. If RAS is configured on the system, all of the RAS IEM (interesting event messages) events are also logged for consumption, as discussed below.

NXD and RAS

IEM is a framework of the RAS feature. Using this framework, NXD logs the NXD IEM events with an application code and its module information:

- The NXD application code is 042
- NXD module codes are:

Code	Module
001	CLI
002	NXD device monitor
003	Core NXD driver

The details of NXD-related RAS IEM codes are:

IEM Event Code (IEC)	Description	Additional Data
042001001	Added Virtual drive to Cache group	VDName, CGName
042001002	Removed Virtual drive to Cache group	VDName, CGName
042001003	Added Cache device to Cache group	CDevName, CGName
042001004	Removed Cache device from Cache group	CDevName, CGName
042001005	Cache state changed	CGStateNew, CGName
042001006	Pinned cache discarded for Virtual drive	CGName, VDName
042001007	Flush parameter settings changed for cache group	CGName, Size, Rate, Interval
042001008	Flush started for Virtual drive	VDName, CGName
042001009	Flush stopped for Virtual drive	VDName, CGName
042001010	Flush started for Cache device	CDevName, CGName
042001011	Flush stopped for Cache device	CDevName, CGName
042001012	Activate completed successfully	CGName

IEM Event Code (IEC)	Description	Additional Data
042001013	Deactivate completed successfully	CGName
042001014	Error CLI	ErrorString
042002001	NytroXD Device Monitor Starting	None
042002002	NytroXD Device Monitor Stopping	None
042002003	NytroXD Device Monitor Cache Device Added	CacheDev
042002004	NytroXD Device Monitor Virtual Drive Added	VirtualDrive
042002005	NytroXD Device Monitor Cache Device Deleted	CacheDev
042002006	NytroXD Device Monitor Virtual Drive Deleted	VirtualDrive
042002007	NytroXD Device Monitor Cache Device Added During StartUp	CacheDev
042002008	NytroXD Device Monitor Virtual Drive Added During StartUp	VirtualDrive
042002009	NytroXD Device Monitor Device State Change From udev	State
042002010	NytroXD Device Monitor Error	Error

14 NXD Support Bundles

The `nxd-get.sh` tool, which is part of the NXD support bundle, collects all the NXD-related log files from each of the OSS nodes. When run on a given OSS node, the tool compresses the NXD-related log files into a tar file named: `nxd-get.`*`hostname`*`.`*`timestamp`*`.tar.gz`

This file is usually located in the `/var/log` directory.

Log File Categories

Captured NXD log files are grouped into categories:

- System files
- System information files
- System log files
- NXD configuration files
- NXD information files
- NXD debug log files

The specific log files in each category and the type of information included in them for each OSS node is listed below. For “standard” log files, like `/var/log/messages`, the details of what to look for to obtain NXD-related log information is also mentioned below.

System Files

`/proc/cmdline`

`/proc/cpuinfo`

`/proc/diskstats`

`/proc/loadavg`

`/proc/mdstat`

`/proc/meminfo`

`/proc/partitions`

`/proc/slabinfo`

`/proc/stat`

`/proc/swaps`

`/proc/uptime`

`/proc/version`

/etc/redhat-release

System Information Files

mpstat.txt	Contains output of `mpstat -A`
sar.txt	Contains output of `sar -d`
dev_mapper_entry.txt	Contains output of `ls -l /dev/mapper/nytro*`
md_by_id_entries.txt	Contains output of `ls -l /dev/disk/by-id/md*`

System Log Files

/mnt/mgmt/var/log/messages	All messages containing either the keyword “NXD” or “ccoh”

NXD Configuration Files

/etc/sba/nytroxd-params.conf

/var/log/nxdlibconf.ini

/opt/seagate/nytroxd/raslog.conf

NXD Information Files

nxd_filter_drv_modinfo.txt

nxd_cache_lib_modinfo.txt

nxd_cli_ver.txt

nxd_show.txt

nxd_show_cachedev.txt

nxd_show_virtualdrive.txt

nxd_show_perfmon.txt

nxd_show_raslogparams.txt

nxd_show_version.txt

nxd_show_logparams.txt

NXD Debug Log Files

/var/log/nxdlibdebug.log