SPECTRA

DEEP STORAGE EXPERTS

Spectra Verde Family Array

Command Line Interface Guide

SpectraLogic.com

Notices

Except as expressly stated herein, Spectra Logic Corporation makes its products and associated documentation on an "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, BOTH OF WHICH ARE EXPRESSLY DISCLAIMED. In no event shall Spectra Logic be liable for any loss of profits, loss of business, loss of use or data, interruption of business, or for indirect, special, incidental or consequential damages of any kind, even if Spectra Logic has been advised of the possibility of such damages arising from any defect or error.

Information furnished in this manual is believed to be accurate and reliable. However, no responsibility is assumed by Spectra Logic for its use. Due to continuing research and development, Spectra Logic may revise this publication from time to time without notice, and reserves the right to change any product specification at any time without notice.

Part Number

90990091 Revision D

Revision History

Revision	Date	Description
A	August 2013	Initial release
B	October 2013	Updated for the Verde 1.3 release.
C	March 2015	Updated for the Verde 2.1 release.
D	May 2016	Updated with new commands.

Note: To make sure you have the most current version of this guide check the Spectra Logic Technical Support portal at support.spectralogic.com/documentation/user-guides/.

To make sure you have the release notes for the most current version of the Verde software, check the Spectra Logic Technical Support portal at support.spectralogic.com/documentation/release-notes/. You must sign into the portal before viewing Release Notes. The release notes contain updates to the *User Guide* since the last time it was revised.

End User License Agreement

You have acquired a Spectra Logic product that includes software owned or licensed by Spectra Logic from one or more software licensors ("Software Suppliers"). Such software products, as well as associated media, printed materials and "online" or electronic documentation ("SOFTWARE") are protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties.

If you do not agree to this end user license agreement (EULA), do not use the Spectra product; instead, promptly contact Spectra Logic for instructions on return of the Spectra product for a refund. Any use of the Software, including but not limited to use on the Spectra product, will constitute your agreement to this EULA (or ratification of any previous consent).

Grant of License. The Software is licensed on a non-exclusive basis, not sold. This EULA grants you the following rights to the Software:

- You may use the Software only on the Spectra product.
- **Not Fault Tolerant.** The Software is not fault tolerant. Spectra Logic has independently determined how to use the Software in the Spectra product, and suppliers have relied upon Spectra Logic to conduct sufficient testing to determine that the Software is suitable for such use.
- **No Warranties for the SOFTWARE.** The Software is provided "AS IS" and with all faults. The entire risk as to satisfactory quality, performance, accuracy, and effort (including lack of negligence) is with you. Also, there is no warranty against interference with your enjoyment of the Software or against infringement. If you have received any warranties regarding the SOFTWARE, those warranties do not originate from, and are not binding on Software suppliers.
- **Note on Java Support.** The Software may contain support for programs written in Java. Java technology is not fault tolerant and is not designed, manufactured, or intended for use of resale as online control equipment in hazardous environments requiring fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communications systems, air traffic control, direct life support machines, or weapons systems, in which the failure of Java technology could lead directly to death, personal injury, or severe physical or environmental damage.
- **No Liability for Certain Damages.** Except as prohibited by law, Software suppliers shall have no liability for any indirect, special, consequential or incidental damages arising from or in connection with the use or performance of the Software. This limitation shall apply even if any remedy fails of its essential purpose. In no event shall Software suppliers, individually, be liable for any amount in excess of U.S. two hundred fifty dollars (U.S. $250.00).
- **Limitations on Reverse Engineering, Decompilation, and Disassembly.** You may not reverse engineer, decompile, or disassemble the Software, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.
- **Software Transfer Allowed with Restrictions.** You may permanently transfer rights under this EULA only as part of a permanent sale or transfer of the Spectra product, and only if the recipient agrees to this EULA. If the Software is an upgrade, any transfer must also include all prior versions of the Software.
- **Export Restrictions.** Export of the Software from the United States is regulated by the Export Administration Regulations (EAR, 15 CFR 730-744) of the U.S. Commerce Department, Bureau of Export Administration. You agree to comply with the EAR in the export or re-export of the Software: (i) to any country to which the U.S. has embargoed or restricted the export of goods or services, which as May 1999 include, but are not necessarily limited to Cuba, Iran, Iraq, Libya, North Korea, Sudan, Syria, and the Federal Republic of Yugoslavia (including Serbia, but not Montenegro), or to any national or any such country, wherever located, who intends to transit or transport the Software back to such country; (ii) to any person or entity who you know or have reason to know will utilize the Software or portion thereof in the design, development or production of nuclear, chemical, or biological weapons; or (iii) to any person or entity who has been prohibited from participating in U.S. export transactions by any federal agency of the U.S. government. You warrant and represent that neither the BXA nor any other U.S. federal agency has suspended, revoked or denied your export privileges.

Contacting Spectra Logic

To Obtain General Information

Spectra Logic Website: www.spectralogic.com

United States Headquarters

Spectra Logic Corporation
6285 Lookout Road
Boulder, CO 80301
USA
Phone: 1.800.833.1132 or 1.303.449.6400
International: 1.303.449.6400
Fax: 1.303.939.8844

European Office

Spectra Logic Europe Ltd.
329 Doncastle Road
Bracknell
Berks, RG12 8PE
United Kingdom
Phone: 44 (0) 870.112.2150
Fax: 44 (0) 870.112.2175

Spectra Logic Technical Support

Technical Support Portal: support.spectralogic.com

United States and Canada

Phone:
Toll free US and Canada: 1.800.227.4637
International: 1.303.449.0160

Europe, Middle East, Africa

Phone: 44 (0) 870.112.2185

Deutsch Sprechende Kunden

Phone: 49 (0) 6028.9796.507
Email: spectralogic@stortrec.de

Mexico, Central and South America, Asia, Australia, and New Zealand

Phone: 1.303.449.0160

Spectra Logic Sales

Website: www.spectralogic.com/shop

United States and Canada

Phone: 1.800.833.1132 or 1.303.449.6400
Fax: 1.303.939.8844
Email: sales@spectralogic.com

Europe

Phone: 44 (0) 870.112.2150
Fax: 44 (0) 870.112.2175
Email: eurosales@spectralogic.com

To Obtain Documentation

Spectra Logic Website: support.spectralogic.com/documentation

Contents

About This Guide

This guide describes how to configure, monitor, and maintain Spectra® Verde®, Verde DPE™, and Verde DP™ arrays, through the command line interface. Unless otherwise noted the instructions in this guide apply to all Verde arrays. The product is referred to as the *Verde family, Verde array,* or the *array* in these instructions.

Intended Audience

This guide is intended for data center administrators and operators who maintain and operate file storage systems. The information in this guide assumes a familiarity with a command line interface, computing terminology, RAID technology, SAS connectivity, and Ethernet networking. You also need to be familiar with installing, configuring, and using data file storage and archival software.

Related Publications

The following documents related to the Spectra Verde family arrays are available on the Support Portal website at support.spectralogic.com, and from the Documentation screen on the Verde web interface.

- The *Spectra Verde Array Family User Guide* provides information about configuring, using and maintaining your Verde family array.
- The *Spectra Verde Array Family Network Setup Tips* provide helpful instructions for troubleshooting common connectivity problems.
- The *Spectra Verde Array Family Quick Start Guide* provides basic instructions for the essential installation and configuration steps.
- The *Spectra Verde Array Family Site Preparation Guide* provides important information that you should know before installing a Verde array in your storage environment.
- The *Spectra Verde Array Family Installation Guide* provides instructions for installing a Verde array.

The following documents are available after logging into your Support portal account at: support.spectralogic.com.

- The *Spectra Verde Family Release Notes and Documentation Updates* provide the most up-to-date information about the Verde arrays, including information about the latest software releases and documentation updates.
- The *Spectra 12- & 36-Drive Chassis Boot Drive Replacement Guide* provides instructions for replacing a failed boot drive in the array.
- The *Spectra 12-, 36- & 45-Drive Chassis Drive Replacement Guide* provides instructions for replacing a failed data drive after the array is installed.
- The *Spectra 12-, 36- & 45-Drive Chassis Fan Replacement Guide* provides instructions for replacing a failed fan in the array.
- The *Spectra 12-, 36- & 45-Drive Chassis Power Supply Replacement Guide* provides instructions for replacing a failed power supply after the array is installed.
- The *Spectra 12-Drive Chassis HBA Replacement Guide* and *Spectra 36-Drive Chassis HBA Replacement Guide* provide instructions for replacing a failed HBA in the array.
- The *Spectra 96-Drive Chassis Drive Replacement Guide* provides instructions for replacing a failed data drive in the Verde DPE expansion node.
- The *Spectra 96-Drive Chassis Fan Replacement Guide* provides instructions for replacing a failed fan in the Verde DPE expansion node.
- The *Spectra 96-Drive Chassis Power Supply Replacement Guide* provides instructions for replacing a failed power supply in the Verde DPE expansion node.

TYPOGRAPHICAL CONVENTIONS

This guide uses the following conventions to highlight important information:

Note: Read text marked by "Note" for additional information or suggestions about the current topic.

Important Read text marked by the "Important" icon for information that helps you complete a procedure or avoid extra steps.

Caution Read text marked by the "Caution" icon for information you must know to avoid damaging the array or disk drives, or losing data.

WARNING Read text marked by the "Warning" icon for information you must know to avoid personal injury.

WARNUNG Lesen Sie markierten Text durch die "Warnung"-Symbol für die Informationen, die Sie kennen müssen, um Personenschäden zu vermeiden.

These release notes use an arrow (···⁘) to describe a series of menu selections. For example:

Select **Configuration** ···⁘ **Network**.

— means —

Select **Configuration**, then select **Network**.

CHAPTER 1

Overview

This chapter describes using the command line interface to configure, use, and maintain a Verde array.

FEATURES OF VERDE FAMILY ARRAYS

The Verde arrays include the following features:

Easy Network-Based Administration The Verde arrays can be configured over an Ethernet network using a standard web browser.

Expansion Nodes The Verde 4U expansion node accommodates up to 44 disk drives with an active bezel, and 45 disk drives with a passive bezel, and can be connected to a master node to increase overall capacity of the array. The Verde DPE expansion node holds up to 96 disk drives. Each expansion node can only be used with its respective master node.

HotPair Two Verde master nodes or two Verde DPE master nodes can be connected to multiple expansion nodes in a failover configuration. One master node acts as the primary controller, and the other acts as the secondary. In the event that the secondary controller detects a failure of the primary controller, it automatically takes over to provide uninterrupted operation, without administrative intervention.

File Sharing Connectivity for Major Operating Systems The Network File System (NFS) and Common Internet File System (CIFS) protocols provide connectivity to most major operating systems, including Microsoft Windows, Apple Macintosh, UNIX, and Linux. Solid state disk drives may be installed in your array to improve NFS performance.

10GBase-T Ethernet Connectivity Two onboard 10 gigabit copper ports (10GBase-T) provide Ethernet connectivity for the arrays with one dedicated port used to access the Verde web interface. If your array contains a 10 GigE Fibre Channel card, the 10GBase-T ports cannot be used for data transfer.

Mirrored Boot Drives Two dedicated, mirrored drives provide the dedicated storage for the operating system.

Network File Interface The Spectra Network File Interface (NFI) feature allows you to automatically move data from your Verde, Verde DPE, or Verde DP array to one or more BlackPearl® gateways, without the need to use a Spectra S3 client. Data is transferred on a schedule and you can configure the array to either keep or delete data after it is copied to the gateway. When a user needs access to data deleted from the Verde array, the BlackPearl gateway copies it back to the array.

Rack-Mount Hardware The Verde arrays are designed to mount in a standard 4-post, 19-inch (48.3 cm) rack using just 2U (3.5 inches, 8.9 cm) or 4U (7 inches, 17.8 cm) of rack space, depending on the size of the array. Each Verde array includes rack mounting hardware. Alternatively, you can place the array on a level tabletop or other horizontal surface.

RAID-Protected Data Disks Data drives in the Verde, Verde DPE, or Verde DP array provide the array's storage capacity. Disk drives are grouped into protected volumes with selectable parity options and automatic data integrity verification to protect against data corruption.

Redundant Components The arrays feature N+1 redundant power supplies and data drives that are hot-swappable for uninterrupted operation. Any data drives not configured in a storage pool act as global spares. A spare becomes active if a drive in a storage pool fails.

Replicated System Configuration The Verde, Verde DPE, and Verde DP master node mirrors its boot drives on to an existing storage pool on the array's data disks. If one or both boot drives fail, the system recovers automatically when replacement boot drives are installed.

Verde Web Interface The Verde web interface is used to perform configuration and management tasks on the Verde arrays. It also lets you monitor the Verde array hardware and view system messages.

ZIL Support The Verde DPE array supports four or six RAID protected solid state drives as a ZIL (ZFS Intent Log). The ZIL drives increase write speed to shared NFS volumes on the array.

10 Gigabit Ethernet A dual port, 10 Gigabit Ethernet (10 GigE) network interface card is installed to provide high-speed data connections between hosts and the Verde array. The 10 GigE card is optional in the Verde 2U master node.

Overview of Storage Pools, Services, Volumes, and Shares

Storage pools, services, volumes, and shares are the logical components used to interact with the data storage capacity provided by the Verde arrays.

Storage Pools

A storage pool is a protected volume, which groups a set of physical drives together to create a virtual drive that the operating system treats as a single physical drive. Depending on how it is configured, a storage pool can provide mirrored, single-parity, double-parity, or triple-parity data protection. Higher levels of protection allow for more individual drives to fail before the data is compromised. The costs of higher protection are reduced storage availability and reduced performance.

Note: Some arrays have restrictions on the stripe size and protection level of pools. See the *Spectra Verde Array Family User Guide* for more information.

Services

Services on the Verde master node are methods of sharing volumes for use by other computers on the network. Currently, the service options available are the CIFS, NFI, NFS, and Replication services. Additionally, the SNMP service is used with network management systems.

Volumes and Shares

Volumes (also known as directories) are located on each storage pool. Volumes can be configured with a minimum and maximum size. When you create a volume, you can specify whether it uses compression, and whether the time stamp for files is updated when the file is read (access time).

After the volume is created, it can be shared (made available for use by other computers on the network). Volumes are shared via the NFS or CIFS service.

MAJOR COMMANDS OF THE COMMAND LINE INTERFACE

This section lists and defines the major commands available from the command line interface. Each command includes a series of subcommands and parameters. Refer to the subsequent chapters in this guide for more detailed information on each major command.

The commands are listed in alphabetical order.

config

The config command has a set of subcommands for configuring many aspects of the system, such as creating storage pools, volumes, shares, configuring mail users, and setting the hostname.

For more information on the config command, see Chapter 3 – config Command on page page 22.

exit

The exit command allows you to log out the current user.

For more information on the exit command, see Chapter 4 – exit Command on page page 77.

help

The help command, used in conjunction with other commands, provides help text and detailed information for each command.

For more information on the help command, see Chapter 5 – help Command on page page 78.

history

The history command provides a list of the last 100 commands run in the command line interface.

For more information on the history command, see Chapter 6 – history Command on page page 79.

network

The network command has a set of subcommands for configuring the Gigabit, 10GBase-T, and 10 GigE ports on the Verde array.

For more information on the network command, see Chapter 7 – network Command on page page 80.

power

The power command controls rebooting or powering off the array.

For more information on the power command, see Chapter 8 – power Command on page page 94.

status

The status command has a set of subcommands for showing the status of all hardware components in the array.

For more information on the status command, see Chapter 9 – status Command on page page 96.

support

The support command has a set of subcommands for configuring and using the support features of the array such as generating and emailing AutoSupport logs to configured mail recipients, or to Spectra Logic Technical Support.

For more information on the support command, see Chapter 10 – support Command on page page 109.

COMMAND SYNTAX

Commands are constructed by using a major command, followed by the desired subcommand. Most subcommands have parameters which configure settings related to the subcommand.

Note: Commands, subcommands, and parameters are case sensitive.

Usage example

```
command subcommand --parameter=value
```

Some parameters, when omitted, default to a specified setting. These default settings are listed in this guide where applicable. If a parameter has no default setting, then omitting the parameter does not configure any setting for that parameter.

For example, when configuring network interface ports, the `mtu` (Mass Transmission Unit) and `gateway` are optional parameters. If the `mtu` parameter is omitted, it defaults to a specified setting. If the `gateway` parameter is omitted, then no gateway is configured.

Most parameter values can be entered into commands without additional characters. However, if you are entering a string of data that contains the space character, you must enter the parameter value inside single quotes.

SCRIPTING

The Verde array command line interface supports scripting, which can be used to automate certain tasks. Certain parameters can be used with commands to modify the command output. The following parameters may be useful when scripting.

Note: Commands that can use these parameters are noted in this guide.

Parameter	Definition
attr=<value>	The attribute name. If specified, the output only displays the attribute name.
csv	Displays CSV (comma-separated value) output.
json	Displays JSON (JavaScript Object Notation) output.
noheader	Turns off display of the header.
noprogress	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.
printatt	Prints the programmatic attribute names.
raw	Displays raw values.
rawcsv	Displays raw CSV output.
xml	Displays raw XML output.

Important If your script is parsing command line output, you must use the 'noprogress' parameter when entering commands via a script.

Scripting Restriction

A user can only access the command line interface on a host machine using an SSH program. Scripting must run from a shell script, Powershell® script, or similar script on a separate machine.

CHAPTER 2

Using the Verde Array

This chapter describes the features of the Verde array and the initial steps you need to take to use the array in your storage environment. For more detailed information about the features and use of the array, see the *Spectra Verde Array Family User Guide*.

Install the Verde Array

You must install additional optional hardware, install the array in a rack or on a sturdy work surface, and power on the array before configuring the array using the command line interface. See the *Spectra Verde Array Family Installation Guide* for complete instructions on installing the array.

Access the Command Line Interface

To access the command line interface, you must use a SSH (Secure Shell) program installed on a host machine with network access to the Verde array. Spectra Logic recommends that the SSH program support scrolling.

1. Launch the SSH application.
2. Enter the IP address of the management port. By default, the login IP address is:
   - **IP Address:** 10.0.0.2
   - **Netmask:** 255.255.255.0

   Note: If your data center already uses this IP address for another device, see the "Troubleshooting" chapter of the *Spectra Verde Array Family User Guide* for instructions on resolving the conflict.
3. After entering the IP address of the management port, a security warning screen may appear. Continue past the warning to access the command line interface.
4. The login prompt displays. Enter the username and password to access the command line interface.

   The default for both the username and password is **spectra**, in all lowercase letters.

   Note: If your SSH program supports it, you can set a hostname for the system and use it in place of the IP address to connect to the command line interface. See config hostname on page 29 for more information.

Initial Steps to Configure the Verde Array

The information in this section covers the initial steps you need to perform to use the array in your storage environment.

Configure Network Settings

Before your backup software can access the Verde array, you must configure the data network ports on the array. The array includes two onboard 10 gigabit copper ports (10GBase-T) with one port dedicated as the management port. Your array may also contain a 10 GigE Ethernet card. You can only create one data connection to the array. You can configure link aggregation of the same type of ports for better performance.

To configure a single data port as the data connection, use the command network datainterface update on page 82.

To configure a link aggregation as the data connection, use the command network datalagg create on page 83.

Create a Storage Pool

When creating a new storage pool, keep the following in mind:

- Each storage pool requires a minimum of one drive. Spectra Logic recommends using eight drives or more in a storage pool to reduce the impact of the overhead. Overhead is the space on the storage pool used to store parity data, and not used for data storage.
- Drives can only be associated with one storage pool. To create a new storage pool using drives that are already configured in an existing storage pool, you must first delete the existing storage pool. You can then create a new storage pool using newly available drives.
- Any drives not configured in storage pools act as global spare drives. If a drive failure occurs on a Verde array, the array immediately activates a global spare. If the global spare is in the same chassis as the storage pool that claimed it, then when you replace the failed drive it becomes the new global spare. If the global spare is in a different chassis than the storage pool that claimed it, then when you replace the failed drive it is rebuilt into the storage pool and the spare drive becomes a global spare again.
- Spectra Logic recommends leaving at least one drive for a global spare.
- The Verde DPE array is limited to a single 23-drive storage pool in a triple parity configuration. The remaining two drives in the Verde DPE array are global spares. If your array configuration includes a Verde DPE expansion node, you can only create additional storage pools in increments of 23 drives.
- The Verde DP array is limited to a single storage pool using all twelve drives in a double or triple parity configuration.

Use the command config pool create on page 48 to create a storage pool.

Create Volumes

Before you begin using a storage pool to store data, you must create one or more volumes to organize how to store data on the pool. After creating a volume, you can share the volume using NFS or CIFS, but not both.

Spectra Logic recommends that you do not create more than 10 volumes. Additional volumes above this limit decrease the performance of the command line interface, as well as the web interface.

Use the command config volume create on page 71 to create volumes.

Configure Sharing

After you create one or more volumes, you can share a volume using either the NFS or CIFS service. Decide which method to use for sharing and use the appropriate command below.

- To create a CIFS share, use the command config cifsshare create on page 26.
- To create an NFS share, use the command config nfsshare create on page 43.

Note: Shares are available after you configure network settings.

CHAPTER 3

config Command

This chapter provides instructions for using the `config` command, and also lists and defines all subcommands of the `config` command.

config

Use the `config` command and its set of subcommands to configure aspects of the system such as creating storage pools, volumes, shares, configuring mail users, and setting the hostname.

All subcommands are listed in alphabetical order.

Note: At any time in the command line interface, you can type `help <command>` for detailed information about the command and any required parameters. You can also use the question mark character '?' in place of `help`.

config asu

Use the `config asu` commands to configure the Automated Software Upload feature, which periodically checks a specified server to determine the availablity of updated software for the array. The feature can also automatically download the updated software package to the array.

config asu list

This command displays the configured settings for the Automated Software Upload feature. This display includes information about the status of the feature, the URL to use to check for updates, proxy information, and the scheduled frequency for checking for updates. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config asu list
```

displays the configured settings for the Automated Software Upload feature.

config asu show

This command displays the configured settings for the Automated Software Upload feature. This display includes the same information as config asu list. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config asu show
```

displays the configured settings for the Automated Software Upload feature.

config asu update

This command configures the Automated Software Upload feature. Use this command in conjunction with the parameters below.

Parameter	Definition
email=<value> (optional)	Configures the Automated Software Upload feature to not automatically download a new software package but instead email the specified address when a new software package is available for manual download.
enabled=<true or **false>** (optional)	Enable the Automated Software Upload feature.
proxy_address=<value> (optional)	The IP address of the proxy server.
proxy_port=<value> (optional)	The port to use on the proxy server.
schedule=<value> (optional)	The frequency that the Automated Software Upload feature checks for software updates. **Values:** Weekly, Every Other Week, Monthly, Every Other Month.
use_proxy=<true or **false>** (optional)	Configures the system to use a proxy server to reach the update server. You must also specify the values for **proxy_port** and **proxy_address**.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config asu update --enabled=true --schedule=weekly
```

enables the Automated Software Upload feature and configures it to check for software updates every week.

config cifsservice

Use the `config cifsservice` commands to display details of the CIFS service on the array, and join, or leave an Active Directory domain.

config cifsservice show

This command displays information about the CIFS service running on the array. The display includes the service status, Active Directory status, and domain name, as well as the array hostname used in the Active Directory domain. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config cifsservice show
```

displays information about the CIFS service configured on the array.

config cifsservice update

This command configures the CIFS service. Use this command in conjunction with at least one of the parameters below.

Parameter	Definition
joindomain (optional)	Join a specified CIFS domain.
leavedomain (optional)	Leaves the CIFS domain to which you are currently joined.
uadomain=<value> (optional)	The name of the domain to join. ▪ If omitted, the interface prompts you for this information. **Note:** Providing information via command parameters is less secure than via a prompt.
uapassword=<value> (optional)	The password to use when joining the domain. ▪ If omitted, the interface prompts you for this information. **Note:** Providing information via command parameters is less secure than via a prompt.
uauser=<value> (optional)	The user name to use when joining the domain. ▪ If omitted, the interface prompts you for this information. **Note:** Providing information via command parameters is less secure than via a prompt.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config cifsservice update --joindomain
--uadomain='Customer Domain' --uapassword=password
--uauser=customer
```

joins a domain called "Customer Domain" using the user name of "customer" and a password of "password".

config cifsshare

Use the `config cifsshare` commands to create and delete CIFS shares, and to view information about each CIFS share configured on the array.

config cifsshare create

This command creates a new CIFS share. Before you can create a CIFS share, you must create a storage pool (see config pool on page 47 for more information), and a volume on the storage pool (see config volume on page 71 for more information). In addition, you must configure your data path (see Chapter 7 – network Command on page 80). Use this command in conjunction with the parameters listed below.

Parameter	Definition
name=<value>	The desired name for the CIFS share, which appears in the Active Directory list of shares.
volume=<value>	The volume name or ID used to create the CIFS share.
path=<value> (optional)	The directory in the volume to share. Enter "/" to share the entire volume.
readonly=<true or **false>** (optional)	Whether the share is created as read only. If **false**, the share is created with full read/write capabilities. ▪ If omitted, this value defaults to 'false'.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config cifsshare create --name=Share1
--volume=Volume1 --path=/
```

creates a CIFS share named “Share1”, on a volume called “Volume1” which shares the entire volume with full read/write capabilities.

config cifsshare delete

This command deletes an existing CIFS share, but does not delete the volume that was using the share, nor any data on that volume. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	Name or ID of the CIFS share you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config cifsshare delete --id=Share1
```

deletes the CIFS share named “Share1”.

config cifsshare list

This command lists all currently configured CIFS shares on the array. The list includes CIFS share ID, name, path, status, read only configuration, and the name of the shared volume. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config cifsshare list
```

displays a list of all CIFS shares configured on the array.

config cifsshare show

This command displays detailed information about the specified CIFS share. This command displays the same information as config cifsshare list on page 27. Use this command in conjunction with the parameters listed below.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The name or ID of the CIFS share to view.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config cifsshare show --name=Share1
```

displays detailed information about the CIFS share named “Share1”.

config clock

Use the `config clock` commands to display and configure the time and date on the array.

config clock show

This command displays the current system time, date, and any NTP servers configured on the array. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config clock show
```

displays the current time and date settings on the array.

config clock update

This command lets you manually configure the time for the array, or enable NTP and configure a list of NTP servers. Use this command in conjunction with at least one of the following parameters.

Parameter	Definition
currenttime=<value> (optional)	Use this parameter to enter time and date settings for the array manually. Use the form: '01/31/2014 15:00' to set the time and date. The clock settings are based on a 24-hour clock. **Note:** This parameter is invalid if NTP is enabled.
ntp=<true or **false>** (optional)	Whether NTP is enabled. If **false**, NTP is disabled.
ntpservers=<value> (optional)	A comma separated list of NTP servers.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config clock update --ntp=true
--ntpservers=0.pool.ntp.org,1.pool.ntp.org
```

enables the array to use NTP and use the NTP servers "0.pool.ntp.org" and "1.pool.ntp.org".

config hostname

Use the `config hostname` commands to set and display the hostname of the array.

config hostname show

This command displays the hostname of the array. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config hostname show
```

displays the hostname of the array.

config hostname update

This command sets the hostname for the array. Use this command in conjunction with at least one of the following parameters.

Parameter	Definition
name=<value> (optional)	The desired name for the host. The hostname displays as the command prompt for the command line interface.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config hostname update --name=Verde1
```

sets the hostname of the array to “Verde1”.

config mailrecipient

Use the `config mailrecipient` commands to create, modify, and delete mail recipients, as well as list the recipients configured on the array, and display detailed information about a specified mail recipient. All log sets and messages are sent to a mail recipient. You cannot send log sets or messages directly to an email address.

config mailrecipient create

This command creates a new mail recipient on the array. Mail recipients can receive system messages when the array generates them, or receive log sets when the array generates them either by schedule, when an error occurs, or both. See the *Spectra Verde Array Family User Guide* for more information about the different types of message severities. Use this command in conjunction with the following parameters.

Parameter	Definition
name=<value>	The desired name for the mail recipient. This name is used internally on the array to identify the mail recipient.
emailaddress=<value>	The email address for this mail recipient. The email address entered is the address the array sends messages and/or log sets to when they are generated.

Parameter	Definition
errorlogsets= **<true** or **false >** (optional)	Whether the mail recipient receives log sets whenever they are generated due to an error on the array. ▪ If omitted, this defaults to 'false'.
scheduledlogsets= **<true** or **false >** (optional)	Whether the mail recipient receives log sets when they are generated on a monthly schedule. ▪ If omitted, this defaults to 'false'.
senderrormsg= **<true** or **false >** (optional)	Whether the mail recipient receives error messages when they are generated by the array. ▪ If omitted, this defaults to 'true'.
sendinfomsg= **<true** or **false >** (optional)	Whether the mail recipient receives information messages when they are generated by the array. ▪ If omitted, this defaults to 'false'.
sendwarnmsg= **<true** or **false >** (optional)	Whether the mail recipient receives warning messages when they are generated by the array. ▪ If omitted, this defaults to 'false'.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config mailrecipient create
--name=VerdeEmail1
--emailaddress=customer@customersite.com
--errorlogsets=true --scheduledlogsets=false
--senderrormsg=true --sendinfomsg=false
--sendwarnmsg=true
```

creates a mail recipient with the internal name of "VerdeEmail1", using "customer@customersite.com" as an email address. The mail recipient receives error log sets, and error and warning messages, but does not receive scheduled log sets or information messages.

config mailrecipient delete

This command deletes a previously configured mail recipient. Use this command in conjunction with the parameters below.

Note: You cannot delete the mail recipient with an email address of autosupport@spectralogic.com.

Parameter	Definition
id=<value>	The name or ID of the mail recipient to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config mailrecipient delete --id=VerdeEmail1
```

deletes the mail recipient with the name “VerdeEmail1”.

config mailrecipient email

This command sends a test email to the specified mail recipient. Use this command in conjunction with at least one of the parameters below.

Parameter	Definition
id=<value> (optional)	The name or ID of the mail recipient to send a test email.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config mailrecipient email --id=VerdeEmail1
```

sends a test email to the mail recipient with the name “VerdeEmail1”.

config mailrecipient list

This command displays a list of all mail recipients configured on the array. The list shows the ID, name, and configured email address of all mail recipients.

Usage example

```
config mailrecipient list
```

displays a list of all configured mail recipients.

config mailrecipient show

This command shows detailed information about a specified mail recipient. This information includes all the settings configured with the config mailrecipient create on page 30. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The name or ID of the mail recipient to display more detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example -

```
config mailrecipient show --id=VerdeEmail1
```

displays detailed information about the mail recipient with the name “VerdeEmail1”.

config mailrecipient update

This command updates an existing mail recipient with the information specified in the command parameters. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name or ID of the mail recipient to update.
name=<value> (optional)	The new desired name for the mail recipient. This name is used internally on the array to identify the mail recipient.
emailaddress=<value> (optional)	The new email address for this mail recipient. The email address entered is the address the array uses to send messages and/or log sets to when they are generated.
errorlogsets= **<true** or **false >** (optional)	Whether the mail recipient receives log sets whenever they are generated due to an error on the array.
scheduledlogsets= **<true** or **false >** (optional)	Whether the mail recipient receives log sets when they are generated on a monthly schedule.
senderrormsg= **<true** or **false >** (optional)	Whether the mail recipient receives error messages when they are generated by the array.
sendinfomsg= **<true** or **false >** (optional)	Whether the mail recipient receives information messages when they are generated by the array.
sendwarnmsg= **<true** or **false >** (optional)	Whether the mail recipient receives warning messages when they are generated by the array.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config mailrecipient update --name=VerdeEmail1
--emailaddress=customer2@customersite.com
--errorlogsets=false --scheduledlogsets=true
--senderrormsg=fase --sendinfomsg=true
--sendwarnmsg=false
```

updates the existing mail recipient with the name of “VerdeEmail1” to use the new email address“customer2@customersite.com”. The mail recipient will not receive error log sets, or error and warning messages, and will receive scheduled log sets, and information messages.

config nfiservice

Use the `config nfiservice` command to display the status of the NFI service.

config nfiservice show

This command shows the status of the NFI service running on the array. This command does not have any parameters.

Usage example

```
config nfiservice show
```

displays information about the NFI service running on the array.

config nfisystem

Use the `config nfisystem` commands to create, modify, and delete the NFI service configuration, as well as list and display detailed information about the NFI service running on the array.

config nfisystem create

This command configures the NFI system running on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
accessid=<value>	The S3 Access ID of a previously created user on the BlackPearl gateway. **Note:** The **secretkey** and **accessid** values must be for the same user.
host=<value>	The hostname of the BlackPearl gateway you want to associate with the NFI service running on the Verde array.
secretkey=<value>	The S3 Secret Key of a previously created user on the BlackPearl gateway. **Note:** The **secretkey** and **accessid** values must be for the same user.
username=<value>	The name of the user associated with the **secretkey** and **accessid** values.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfisystem create
--accessid=Y2xpZmZo --host=blackpearl1
--secretkey=qNG5zB2T --username=bobsmith
```

configures the NFI service on the Verde array to connect to the BlackPearl gateway with a system name of “blackpearl1”, using the user “bobsmith” and the S3 credentials associated with that user.

config nfisystem delete

This command deletes the configuration of the NFI service running on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name or ID of the NFI service to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfisystem delete --id=1
```

deletes the configuration of the NFI service with an ID of “1” on the Verde array.

config nfisystem list

This command displays the configuration of all the NFI configurations on the array. This command displays the ID, host address, username, and S3 Access ID and Secret Key.

Usage example

```
config nfisystem list
```

displays a list of all NFI configurations.

config nfisystem show

This command displays the settings of the specified NFI configuration on the array. This command displays the ID, host address, username, and S3 Access ID and Secret Key. This command displays the same information as config nfisystem list on page 36. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The name or ID of the NFI service to display.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example -

```
config nfisystem show --id=1
```

displays detailed information about the NFI configuration with the ID "1".

config nfisystem update

This command updates an existing NFI configuration on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name or ID of the NFI service to update.
accessid=<value> (optional)	The S3 Access ID of a previously created user on the BlackPearl gateway. **Note:** The **secretkey** and **accessid** values must be for the same user.
host=<value> (optional)	The hostname of the BlackPearl gateway you want to associate with the NFI service running on the Verde array.
secretkey=<value> (optional)	The S3 Secret Key of a previously created user on the BlackPearl gateway. **Note:** The **secretkey** and **accessid** values must be for the same user.
username=<value> (optional)	The name of the user associated with the **secretkey** and **accessid** values.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfisystem update --ID=1 --username=customer2
--accessid=NHUyNS1y --secretkey=p58mhwBD
```

updates the existing NFI configuration with the ID of "2" to use a new user with the name "customer2", an access ID of "NHUyNS1y", and a secret key of "p58mhwBD".

config nfivolumepolicy

Use the `config nfivolumepolicy` commands to modify and display detailed information about the NFI volume policies configured on the array.

config nfivolumepolicy list

This command displays the settings of all the NFI volume policies configured on the array. This command displays the NFI volume policy ID, the status of the volume policy, the ID of the volume used by the NFI service, the name of the bucket on the target BlackPearl gateway, the policy type, and the number of days to retain data on the Verde array, if the policy type is **copy and delete**.

Usage example

```
config nfivolumepolicy list
```

displays the settings of all the NFI volume policies configured on the array.

config nfivolumepolicy show

This command displays the configuration of the specified NFI volume policy on the array. This command displays the same information as config nfivolumepolicy list. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The ID of the NFI volume policy to display.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfivolumepolicy show --id=4
```

displays detailed information about the NFI volume policy with the ID "4".

config nfivolumepolicy update

This command updates an existing NFI volume policy configured on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the NFI volume policy to update.
bucket=<value> (optional)	The name of the bucket on the BlackPearl gateway to which files are transferred from the Verde array.
enabled=<true or **false>** (optional)	Whether or not the volume policy is enabled.
nfisystem=<value> (optional)	The NFI system name to which the files are transferred.
policy=<value> (optional)	The type of volume policy to use when transferring data. **Values:** copy_and_keep, copy_and_delete
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfivolumepolicy update --ID=4
--policy=copy_and_keep
```

updates the existing NFI volume policy with the ID of "4" to use the policy type "copy_and_keep".

config nfivolumepolicyschedue

Use the `config nfivolumepolicyschedule` commands to modify and display detailed information about the NFI volume policy schedules configured on the array.

config nfivolumepolicyschedule list

This command displays the settings of all the NFI volume policy schedules configured on the array. This command displays the NFI volume policy schedule ID, the NFI volume policy ID, and the set schedule when data is transferred. This command has no parameters.

Usage example

```
config nfivolumepolicyschedule list
```

displays the settings of all the NFI volume policy schedules configured on the array.

config nfivolumepolicyschedule show

This command displays the configuration of the specified NFI volume policy schedule on the array. This command displays the same information as config nfivolumepolicyschedule list. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The ID of the NFI volume policy schedule to display.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example -

```
config nfivolumepolicyschedule show --id=f788755a-0027-
11e6-b004-0cc47a34c948
```

displays detailed information about the NFI volume policy schedule with the ID “f788755a-0027-11e6-b004-0cc47a34c948”.

config nfivolumepolicyschedule update

This command updates an existing NFI volume policy schedule configured on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the NFI volume policy schedule to update.
daysofweek=<value> (optional)	A comma separated list of days to copy data to the BlackPearl gateway (for example: 'monday,tuesday') **Note:** Only applies if **recurrence** is set to **weekly**.
every=<value> (optional)	A numerical value specifying the frequency of NFI transfers. ▪ If the volume policy schedule **recurrence** is set to **hourly**, this value specifies that data is transferred every *X* hours. ▪ If the volume policy schedule **recurrence** is set to **daily**, this value specifies that data is transferred every *X* days.
recurrence=<value> (optional)	The recurrence of NFI transfers. **Values:** hourly, daily, weekly
starthour=<value> (optional)	The starting hour for NFI transfers. **Values:** 0-23 **Note:** Only applies if **recurrence** is set to **daily** or **weekly**.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfivolumepolicyschedule update --ID=f788755a-
0027-11e6-b004-0cc47a34c948
--recurrence=weekly
--daysofweek='monday,wednesday,friday' --starthour=20
```

updates the existing NFI volume policy schedule with the ID of "f788755a-0027-11e6-b004-0cc47a34c948" to have a recurrence of "weekly" and to transfer data on the days "Monday, Wednesday, Friday" at a time of "20-hundred hours / 8pm".

config nfsservice

Use the `config nfsservice` commands to configure the NFS service, as well as to show detailed information about the NFS service.

config nfsservice show

This command displays detailed information about the NFS service configured on the array. The information includes the status of the service, if TCP or UDP are enabled, and the number of threads the service is configured to use. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config nfsservice show
```

displays the configured parameters of the NFS service.

config nfsservice update

This command configures the NFS service on the array. Use this command in conjunction with at least one of the following parameters.

Parameter	Definition
tcp=<true or **false >** (optional)	Whether TCP (Transmission Control Protocol) is enabled for the NFS service.
threads=<value> (optional)	The number of threads the NFS service uses. Recommended values are between 64 and 512.
udp=<true or **false >** (optional)	Whether UDP (User Datagram Protocol) is enabled for the NFS service.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfsservice update --tcp=true --threads=256
--udp=false
```

configures the NFS service to enable TCP, to use 256 threads, and to disable UDP.

config nfsshare

Use the `config nfsshare` commands to create, modify, and delete NFS shares, as well as display detailed information about any NFS shares configured on the array.

config nfsshare create

This command creates an NFS share on the array. Before you can create an NFS share, you must create a storage pool (see config pool on page 47) and a volume on the storage pool (see config volume on page 71). In addition, you must configure your data path (see Chapter 7 – network Command on page 80). Use this command in conjunction with the following parameters.

Parameter	Definition
accesscontrol=<value >	A comma separated list of access control entries to be granted permission to access the NFS share, in single quotes. The access control entries can be entered in any of the following ways: ▪ An IP address of a single host (`10.2.1.22`) ▪ A subnet of hosts (`10.2.1.1/24`) ▪ All hosts (`*`) Each access control entry is followed by the permission level desired for that entry. Use one of the following permission levels: ▪ `norootsquash` - This user has root access to the share. This user is used to set permissions for rootsquash users. ▪ `rootsquash` - This user has standard access to the share. ▪ `ro` - Read only. The user can access the share, but cannot write data.
volume=<value>	The volume name or ID to use when creating an NFS share.
comment=<value> (optional)	A comment for the NFS share, if desired. This comment is internal to the Verde software and is not displayed to any hosts accessing the share.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfsshare create --accesscontrol='10.1.2.100
norootsquash, 10.1.1.1/24 rootsquash'
--volume=Accounting --comment=NFS share for Accounting
```

configures an NFS share granting root access to a single host with an IP address of 10.1.2.100, and granting standard access to all hosts on the subnet of 10.1.1.1/24. This share is created using the "Accounting" volume on the array, and has a comment of "NFS share for Accounting".

config nfsshare delete

This command deletes a previously configured NFS share on the array. This command only deletes the NFS share. The command does not delete the volume or the data it contains. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The volume name or ID of the NFS share you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfsshare delete --id=Accounting
```

deletes the existing NFS share with the name "Accounting".

config nfsshare list

This command lists all NFS shares currently configured on the array. The list includes the volume name, ID, status, and comment field. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config nfsshare list
```

lists all NFS shares configured on the array.

config nfsshare show

This command displays detailed information about the specified NFS share. This information includes the volume status, ID, name, comment field, and an access control list. Use this command with the following parameter.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The ID of the NFS share for which you want to view detailed information.

Usage example

```
config nfsshare show --id=8e88e732
```

displays detailed information about the NFS share with the ID “8e88e732”.

config nfsshare update

This command updates an existing NFS share on the array. You can change the internal share comment, or enter a new list of access control entries. The list of hosts allowed permission to the NFS share entered with this command replaces the current list of hosts, making it possible to stop hosts from accessing the share by omitting them from the new access control list. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The volume name or ID to update.
accesscontrol=<value > (optional)	A comma separated list of access control entries to be granted permission to access the NFS share, in single quotes. The access control entries can be entered in any of the following ways: ▪ An IP address of a single host (`10.2.1.22`) ▪ A subnet of hosts (`10.2.1.1/24`) ▪ All hosts (`*`) Each access control entry is followed by the permission level desired for that entry. Use one of the following permission levels: ▪ `norootsquash` - This user has root access to the share. This user is used to set permissions for rootsquash users. ▪ `rootsquash` - This user has standard access to the share. ▪ `ro` - Read only. The user can access the share, but cannot write data.
comment=<value> (optional)	A comment for the NFS share, if desired. This comment is internal to the Verde software and is not displayed to any hosts accessing the share.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config nfsshare update --id=Accounting --comment='NFS
share for Accounting and Sales'
```

configures the existing NFS share of "Accounting" to use the new comment of "NFS share for Accounting and Sales".

config password

Use the `config password` command to update the password of existing users configured on the array.

Note: At this time, only the user named 'spectra' is configured on the array. It is not possible to configure additional accounts.

config password update

This command updates the password of the user named 'spectra'. The interface prompts you to enter the existing password, then enter the new password, and finally to confirm the new password. Use this command with the following parameters.

Parameter	Definition
id=<value>	The ID of the account for which you want to change the password.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config password update --id=1
```

starts the process to change the password for the account with the ID "1". The interface prompts you to enter the current password, the desired new password, and to confirm the new password.

config pool

Use the `config pool` commands to create, modify, expand, and delete storage pools, as well as list the pools configured on the array, and display detailed information about a specified storage pool.

Note: Some arrays require a specific stripe size for a pool. See the *Spectra Verde Array Family User Guide* for restrictions.

config pool canceldataintegrityverification

This command cancels a data integrity verification scan that is currently running. To start data integrity verification, see config pool startdataintegrityverification on page 53.

Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the storage pool for which you want to cancel a data integrity verification scan.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config pool canceldataintegrityverification --ID=Pool1
```

cancels the data integrity verification running on “Pool1”.

config pool create

This command creates a storage pool on the array. Pools are created using a specified amount of disk drives, and configured to use a specified protection level and a specified optimization scheme. Pools can be expanded after they are created (see config pool expand on page 51). Use this command in conjunction with the following parameters.

Parameter	Definition
name=<value>	The desired name for the storage pool.
highwatermark=<value> (optional)	The pool filled capacity percentage that triggers the array to send a message. **Values:** percentage between 0 and 100 ▪ If set to zero, no notification is sent. ▪ If omitted, this value defaults to 80%.

Parameter	Definition
optimization=<value> (optional)	The optimization scheme for the pool. **Values:** ▪ `capacity`—configure the storage pool for maximum capacity. ▪ `performance`—configures the storage pool for maximum performance with increased fault tolerance. ▪ `balance`—balance between the capacity and performance schemes. ▪ `custom` — you are prompted to select a valid RAID configuration. ▪ If omitted, this value defaults to 'balance'. **Note:** Some storage pool configurations do not support this setting.
protection=<value> (optional)	The protection level for the storage pool. **Note:** See the *Spectra Verde Array Family User Guide* for restrictions on parity selection. **Values:** ▪ `none`—create a storage pool with no protection level. Any drive failure results in data loss. ▪ `mirror`—create a storage pool with data striped across two mirrors. This type of RAID offers the best performance for small random reads and writes. ▪ `single`—create a single-parity array, which can tolerate one drive failure without data loss. This type of RAID has faster performance than double and triple parity based RAIDs. ▪ `double`—create a double-parity array, which can tolerate two drive failures without data loss. In most cases, double-parity provides the best balance between data protection, performance, and storage capacity. ▪ `triple`—create a triple-parity array, which can tolerate three drive failures without data loss. This type of RAID provides the most data protection. ▪ If omitted, this value defaults to 'double'.
uadriveids=<value> (optional)	A semicolon-separated list of the disk drive IDs to use when creating a storage pool. To see the ID of each drive, run the status drive list on page 100. **Note:** Some arrays require a specific stripe size for a pool. See the *Spectra Verde Array Family User Guide* for restrictions. ▪ If omitted, the interface prompts you for this information after running the command.

Parameter	Definition
usealldisks (optional)	Configures the storage pool to use all unused disk drives installed in the array.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config pool create --name=Pool1 --highwatermark=75
--protection=triple
```

creates a storage pool with the name “Pool1”, a high watermark percentage of 75 and a protection level of triple-parity. The user will be prompted for drive ids as the **uadriveids** parameter was omitted from the command. The pool uses the optimization scheme of “balance” as the **optimization** parameter was omitted from the command and was configured to use a default setting.

config pool delete

This command deletes a previously configured storage pool from the array.

Caution When you delete a storage pool, all data on it is lost. If you want to keep the data, migrate it to another location before deleting the pool.

Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The volume name or ID of the storage pool you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config pool delete --id=Pool1
```

deletes the existing storage pool with the name “Pool1”. All data on the pool is lost.

config pool deleteall

This command deletes all storage pools configured on the array.

Caution When you delete a storage pool, all data on it is lost. If you want to keep the data, migrate it to another location before deleting the pool.

Use this command in conjunction with the following parameters.

Parameter	Definition
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config pool deleteall
```

deletes all storage pools configured on the array. All data on all pools is lost.

config pool expand

This command expands an existing storage pool to use additional disks present in the array. This command is useful if you recently purchased additional disk drives and want to include them in an existing storage pool.

You can only expand an existing pool by it’s stripe size. For example, if you have a stripe size of 4, you can only expand the pool in multiples of 4 drives. For mirrored volumes, you can only expand the pool in multiples of 2. Any number of drives can be used to expand a pool with no protection. Use the command config pool show on page 53 to determine the stripe size of a pool. If you do not specify the correct number of disks, the command fails and the stripe size for the pool you are trying to expand displays.

Use the command in conjunction with the following parameters.

Parameter	Definition
name=<value>	The name or ID of the storage pool you want to expand.
uadriveids=<value> (optional)	A semicolon-separated list of the disk drive IDs to use when creating a storage pool. To see the ID of each drive, run the status drive list on page 100. ▪ If omitted, the interface prompts you for this information after running the command.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config pool expand --id=Pool1
--uadriveids='ID25;ID26;ID27;ID28'
```

expands the existing storage pool with the name "Pool1" using the disk drives installed in positions 25 through 28.

config pool list

This command lists all storage pools currently configured on the array. The list includes the pool name, ID, status, used and available space, and the configured data protection for the pool. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config pool list
```

displays all storage pools configured on the array.

config pool show

This command shows detailed information about a specified storage pool. The display includes the information listed in config pool list on page 52, and also shows the high water mark percentage, the number of drives, and the number of RAID stripes. Use this command in conjunction with the following parameter.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The name or ID of the storage pool for which you want to view detailed information.

Usage example

```
config pool show --id=Pool1
```

displays detailed information about the storage pool named "Pool1".

config pool startdataintegrityverification

The Verde arrays feature an on-demand data integrity verification for data drives configured in a storage pool. The data integrity verification scans the drives for data corruption and corrects any errors found. You can run a data integrity verification on a single pool, or multiple pools simultaneously.

Notes:
- The transfer speed performance of the data path is reduced when running a data integrity verification.
- Data integrity verifications can be cancelled but cannot be paused. If you cancel a data integrity verification, you must restart the verification from the beginning if you wish to continue scanning the data drives in the pool for corruption.

This command starts a data integrity verification scan on a specified storage pool. The verification scans the drives in the specified storage pool for data corruption and corrects any errors found. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the storage pool for which you want to start data integrity verification.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config pool startdataintegrityverification --id=Pool1
```

starts a data integrity verification scan on the storage pool with the name "Pool1".

config pool update

This command changes the configuration of an existing storage pool. At this time, the only aspect of a storage pool that you can change after it is created is the high watermark setting. To change any other settings of an existing storage pool, you need to delete the existing pool and create a new pool with the desired configuration.

Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name or ID of the storage pool you want to update.
highwatermark=<value> (optional)	The pool filled capacity percentage that triggers the array to send a message. **Values:** a percentage between 0 and 100 ▪ If set to zero, no notification is sent.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config pool update --id=Pool1 --highwatermark=90
```

changes the high watermark setting on the pool named "Pool1" to 90%.

config replicationservice

Use the `config replicationservice` command to display information about the replication service configured on the array.

config replicationservice show

This command displays the status of the replication service running on the Verde array. There are no parameters for this command.

Usage example

```
config replicationservice show
```

displays the status of the replication service running on the Verde array.

config replicationvolumepolilcy

Use the `config replicationvolumepolicy` commands to update an existing replication volume policy, and to display information about the replication volume policies configured on the array.

Note: You cannot create new replication volume policies through the command line interface. You must use the Verde web interface to create a new replication volume policy. See "Configure Replication" in the *Spectra Verde Array Family User Guide.*

config replicationvolumepolicy list

This command lists all replication volume policies present on the array. The list shows the replication volume policy ID, the replication volume policy status, the replication system ID, the ID of the volume being replicated, the target storage pool name, and target volume name. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config replicationvolumepolicy list
```

lists all of the replication volume policies configured on the array.

config replicationvolumepolicy show

This command displays information about a specified replication volume policy configured on the array. This command displays the same information as config replicationvolumepolicy list. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the volume replication policy you want to display.

Parameter	Definition
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config replicationvolumepolicy show --id=4
```

displays information about the replication volume policy with the ID “4”.

config replicationvolumepolicy update

This command updates a specified replication volume policy configured on the array. Use the command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the volume replication policy you want to update.
destinationpoolname=<value> (optional)	The name of the destination storage pool on the target array to which the source volume is replicated.
destinationvolumename=<value> (optional)	The name of the destination volume on the target array to which the source volume is replicated.
enabled=<true or **false>** (optional)	Whether or not the replication volume policy is enabled.
replicationsystemname=<value> (optional)	The replication system name to which the source files are replicated.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config replicationvolumepolicy update --id=4
--enabled=false
```

disables the replication volume policy with the ID “4”.

config replicationvolumepolicyschedule

Use the `config replicationvolumepolicyschedule` commands to update an existing replication volume policy schedule, and to display information about the replication volume policy schedules configured on the array.

Note: You cannot create new replication volume policy schedules through the command line interface. You must use the Verde web interface to create a new replication volume policy schedule. See “Configure Replication” in the *Spectra Verde Array Family User Guide.*

config replicationvolumepolicyschedule list

This command lists all replication volume polcies present on the array. The list shows the replication volume policy schedule ID, the replication volume policy ID, and the date and time of the configured replication schedule. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config replicationvolumepolicyschedule list
```

lists all of the replication volume policy schedules configured on the array.

config replicationvolumepolicyschedule show

This command displays information about a specified replication volume policy schedule configured on the array. This command displays the same information as config replicationvolumepolicyschedule list. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the volume replication policy schedule you want to display.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config replicationvolumepolicy show --id=f84b6448-0027-11e6-b004-0cc47a34c948
```

displays information about the replication volume policies with the ID "f84b6448-0027-11e6-b004-0cc47a34c948".

config replicationvolumepolicyschedule update

This command updates a specified replication volume policy schedule configured on the array. Use the command in conjuction with the following parameters.

Parameter	Definition
id=<value>	The ID of the replication volume policy schedule to update.
daysofweek=<value> (optional)	A comma separated list of days to copy data to the target Verde array (for example: 'monday,tuesday') **Note:** Only applies if **recurrence** is set to **weekly**.
every=<value> (optional)	A numerical value specifying the frequency of replication transfers. ▪ If the volume policy schedule **recurrence** is set to **hourly**, this value specifies that data is transferred every *X* hours. ▪ If the volume policy schedule **recurrence** is set to **daily**, this value specifies that data is transferred every *X* days.
recurrence=<value> (optional)	The recurrence of replication transfers. **Values:** hourly, daily, weekly
starthour=<value> (optional)	The starting hour for replication transfers. **Values:** 0-23 **Note:** Only applies **recurrence** is set to **daily** or **weekly**.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config replicationvolumepolicyschedule update
--id=f84b6448-0027-11e6-b004-0cc47a34c948
--recurrence=daily --starthour=12
```

updates the replication volume policy schedule with the ID "f84b6448-0027-11e6-b004-0cc47a34c948" to replicate data daily, with a start hour time of 12 hundred hours / 12 noon.

config snapshot

Volume snapshots are images of a volume's configuration and data makeup as they were when the array generated the snapshot. Restoring to a previously created snapshot allows you to go "back in time" and restore the volume to the state it was in when the array created the snapshot. You can use snapshots to restore a file that was accidentally deleted. You can create snapshots manually or on a schedule. The array retains volume snapshots until they are deleted.

Use the `config snapshot` commands to create, and delete snapshots, as well as to restore a volume using an existing snapshot. The command also lists all snapshots and allows you to view detailed information about a specified snapshot.

config snapshot create

This command creates a snapshot of an existing volume. Use this command in conjunction with the following parameters.

Parameter	Definition
name=<value> (optional)	The name of the snapshot you want to create. ▪ If omitted, this value defaults to "manual_<*timestamp*>".
volume=<value>	The name or ID of the volume to use when creating a snapshot.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snapshot create --name=Accounting060813
--volume=Accounting
```

creates a snapshot of the volume named “Accounting”. The snapshot is named “Accounting060813”.

config snapshot delete

This command deletes a previously created snapshot on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID or name of the snapshot you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snapshot delete --id=Accounting060813
```

deletes the snapshot named “Accounting060813”.

config snapshot list

This command lists all snapshots present on the array. The list shows the snapshot name, ID, the volume that was used to create the snapshot, and the date and time of the snapshot creation. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config snapshot list
```

lists all of the snapshots present on the array.

config snapshot rollback

This command uses a previously created snapshot to restore a volume's state to when the the array created the snapshot.

Caution Any changes or snapshots created after the snapshot used to restore the volume's state are permanently lost.

Note: For instructions on restoring an individual file, see the *Spectra Verde Array Family User Guide*.

Use this command with the following parameters.

Parameter	Definition
id=<value>	The name of the snapshot you want to use to restore a volume to the state it was in when the snapshot was taken.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snapshot rollback --id=Accounting060813
```

uses the snapshot named "Accounting060813" to restore the volume the snapshot was associated with when it was created.

config snapshot show

This command shows detailed information for the specified snapshot including the snapshot name, ID, the volume that was used to create the snapshot, and the date and time when the array created the snapshot. Use this command in conjunction with the following parameter.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The name or ID of the snapshot for which you want to view more detailed information.

Usage example

```
config snapshot show --id=Accounting060813
```

displays detailed information about the snapshot named "Accounting060813".

config snapshotschedule

Use the `config snapshotschedule` command to create a schedule that controls when the array creates snapshots. Snapshot schedules can be hourly, daily, or weekly.

config snapshotschedule create

This command creates a schedule for when the array creates snapshots of a specified volume. Use this command in conjunction with the following parameters.

Parameter	Definition
name=<value>	The name of the snapshot schedule you want to create.
recurrence=<value>	The recurrence of the snapshot schedule. **Values:** hourly, daily, weekly
volume=<value>	The name or ID of the volume to use when creating a snapshot.
daysofweek=<value>	A comma-separated list of the days of the week that a snapshot is generated. **Note:** Only applies when the **recurrence** is set to **weekly**.
every=<value>	The interval number between snapshot generation. Applies to hourly and daily schedules. ▪ If the **recurrence** is set to **hourly**, this setting configures the array to generate a snapshot every *X* hours. ▪ If the **recurrence** is set to **daily**, this setting configures the array to generate a snapshot every *X* days.
maxnumbersnapshots=<value>	The maximum number of snapshots this schedule retains. Once the maximum number is reached, the oldest snapshot is deleted.

Parameter	Definition
starthour=<value>	The hour of the day that the schedule generates a snapshot. **Values:** 0-23 **Note:** Only applies **recurrence** is set to **daily** or **weekly**.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snapshotschedule create --id=AccountingSchedule
--volume=Accounting --recurrance=daily
--every=3 --maxnumbersnapshots=50
```

creates a snapshot schedule named "AccountingSchedule", using the volume named "Accounting". The snapshot schedule has a recurrence of "daily" and creates a snapshot every 3 days. The snapshot schedule retains 50 snapshots.

config snapshotschedule delete

This command deletes a previously created snapshot schedule on the array. The snapshots created with the schedule you are deleting remain on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name of the snapshot schedule you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snapshotschedule delete --id=AccountingSchedule
```

deletes the snapshot schedule named "AccountingSchedule".

config snapshotschedule list

This command lists all snapshot schedules present on the array. The list includes the schedule name, ID, the volume used to create the snapshots, and the details of the snapshot schedule. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config snapshotschedule list
```

lists all snapshot schedules on the array.

config snapshotschedule show

This command displays detailed information about a specified snapshot schedule. This display includes the information listed in config snapshotschedule list on page 64, and also shows the maximum number of snapshots the schedule maintains before it deletes the oldest snapshot. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Parameter	Definition
id=<value>	The name of a specific snapshot schedule for which you want to view the details.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snapshotschedule show --id=AccountingSchedule
```

shows detailed information for the snapshot schedule named "AccountingSchedule".

config snmphost

Use the `config snmphost` command to configure the array as an SNMP host that can send and receive SNMP queries, notifications, and traps.

config snmphost create

This command configures an SNMP host on the array. You can configure multiple SNMP hosts on the array by running this command multiple times. Use the command in conjunction with the following parameters.

Parameter	Definition
host=<value>	The name or IP address of the SNMP host you want to create.
communitystring=<value> (optional)	The community string the SNMP host uses for communication. Any incoming SNMP queries that use a different community string than the one set here fail. ▪ If omitted, this value defaults to 'public'.
notifications=<true or **false>** (optional)	Whether the SNMP client should be sent outgoing notifications. ▪ If omitted, this value defaults to 'true'.
port=<value> (optional)	The port used for SNMP communication to the array. ▪ If omitted, this value defaults to 162.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snmphost create --host=Verde1
--communitystring=verde --notifications=true
--port=6000
```

configures an SNMP host on the array with the name "Verde1" that uses the community string "verde". The host sends outgoing notifications to clients, and uses port 6000.

config snmphost delete

This command deletes an SNMP host configured on the array. Use this command with the following parameters.

Parameter	Definition
id=<value>	The name of the SNMP host you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snmphost delete --host=Verde1
```

deletes the SNMP host on the array with the name "Verde1".

config snmphost list

This command lists all SNMP hosts that are configured on the array. The list includes the hostname, the port number, and if the host is configured to send notifications. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 14 for more information.

Usage example

```
config snmphost list
```

lists all the SNMP hosts configured on the array.

config snmphost show

This command displays detailed information about the specified SNMP host. The display includes the information listed in config snmphost list, and also shows the community string configured for the SNMP host. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name of a specific SNMP host for which you want to view detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snmphost show --id=Verde1
```

displays detailed information about the SNMP host named “Verde1”.

config snmphost update

This command updates an existing SNMP host on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name of the existing SNMP host you want to update.
host=<value> (optional)	The name or IP address of the SNMP host you want to update.
communitystring=<value> (optional)	The community string the SNMP host uses for communication. Any incoming SNMP queries that use a different community string than the one set here fail.
notifications= **<true** or **false>** (optional)	Whether the SNMP client is sent outgoing notifications.
port=<value> (optional)	The port used for SNMP communication to the array.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snmphost update --host=Verde1
--communitystring=spectra --notifications=false
```

configures the existing SNMP host with the name “Verde1” to use the community string “spectra”. The host does not send outgoing notifications to clients.

config snmpservice

Use the `config snmpservice` commands to configure the SNMP service, as well as to show detailed information about the SNMP service.

config snmpservice show

This command displays the status of the SNMP service and its current configuration. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
config snmpservice show
```

displays the status and configuration of the SNMP service.

config snmpservice update

This command updates the SNMP service configuration. Use this command in conjunction with at least one of the following parameters.

Parameter	Definition
communitystring=<value> (optional)	The community string the SNMP host uses for communication. Any incoming SNMP queries that use a different community string than the one set here fail. .
contact=<value> (optional)	The contact information for the administrator of the Verde array. It is useful to enter the administrators name and phone number.
location=<value> (optional)	The physical location of the Verde array in your company.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config snmpservice update --communitystring=verde
--contact='Desmond Hume, 481-516-2342'
--location='Swan Station'
```

configures the SNMP service to use the community string "verde" and the contact and location information specified in the command.

config summary

Use the `config summary` command to list a summary of the configuration of storage pools and volumes present on the array. It also lists a summary of the NTP, CIFS, and SMNP services. Use this command with the following parameter.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config summary
```

lists the summary of the configuration on the array. The output of this command is the same as running the `list` commands for storage pools, volumes, and the NFS, CIFS, NFI, Replication, and SNMP services.

config user

Use the `config user` commands to modify existing users, as well as list all the users configured on the array, and display detailed information about a specified user.

Note: At this time, only the user named 'spectra' is configured on the array. It is not possible to configure additional accounts.

config user list

This command displays a list of all users configured on the array and displays the user ID, the user's full name, the username, and the role of the user. This command has no parameters.

Usage example

```
config user list
```

displays all the users configured on the array.

config user show

This command displays detailed information about the specified user. This command displays the same information as config user list, as well as displaying the session timeout value, and the status of the remote support feature. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the user you want to display.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config user show --id=1
```

displays information about the user with the ID “1”

config user update

This command updates an existing user configured on the array. This command allows you to set the session timeout value, or to enable or disable the remote support feature. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the user you want to modify.
remotesupportenabled=<true or **false>** (optional)	Enable or disable the remote support function. **Note:** You must have a remote support activation key entered before you can enable or disable the remote support feature. See support activationkey on page 110 for more information.
sessiontimeout=<value> (optional)	The session timout value for the CLI in minutes.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config user update --id=1 --sessiontimeout=60
```

modifys the user with the ID “1” to have a session time out of “60 minutes”

config volume

Use the `config volume` commands to create, modify, and delete volumes, as well as list all the volumes configured on the array, and display detailed information about a specified volume.

config volume cancelmove

This command cancels moving a volume from one storage pool to another. Use the command config volume startmove on page 74 to initiate a volume move. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the volume for which you want to cancel a move to a different storage pool.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config volume cancelmove --ID=Volume1
```

cancels the move in progress of a volume with the name of “Volume1”.

config volume create

This command creates a new volume on an existing storage pool on the array. After you create one or more volumes, share the volumes using either the NFS or CIFS service.

- See config cifsshare on page 26 for information on CIFS shares.
- See config nfsshare on page 43 for information on NFS shares.

Use this command in conjunction with the following parameters.

Parameter	Definition
name=<value>	The desired name for the volume.

Parameter	Definition
pool=<value>	The name or ID of the storage pool on which to create the volume.
atime=<true or **false>** (optional)	Whether the access time attribute for files is updated when they are read. If false, this avoids producing write traffic while reading files and can improve performance. ▪ If omitted, this value defaults to 'false'.
compression=<true or **false>** (optional)	Whether the array compresses data written to the volume. ▪ If omitted, this value defaults to 'false'.
minsize=<value> (optional)	Capacity in bytes. You can enter values in KB, MB, GB, and TB. This space is allocated immediately if sufficient space is available. If insufficient space is available, then volume creation fails. If set to zero, no space is pre-allocated. ▪ If omitted, this value defaults to 'zero'.
maxsize=<value> (optional)	Capacity in bytes. You can enter values in KB, MB, GB, and TB. If set to zero, no maximum size is configured. ▪ If omitted, this value defaults to 'zero'. ▪ Volumes are thin provisioned, so it is possible for the combined allocated maximum storage of all volumes to exceed the size of the storage pool.
readonly=<true or **false>** (optional)	Whether the volume is configured as read only and no data can be written to, or modified, on the volume. ▪ If omitted, this value defaults to 'false'.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config volume create --name=AccountingShare
--pool=Pool1 --compression=true --minsize=100GB
--maxsize=3TB --readonly=false
```

creates a new volume with the name "AccountingShare", on a storage pool with the name "Pool1". This volume uses compression, and has a minimum size of 100 GB, with a maximum size of 3 TB. This volume is configured with full access. This volume does not update time attributes when files are read, as the **atime** parameter was omitted.

config volume delete

This command deletes an existing volume on the array.

Caution When you delete a volume, all data on it is lost. If you want to keep the data, migrate it to another location before deleting the volume.

Use this command with the following parameters.

Parameter	Definition
id=<value>	The name of the volume you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config volume delete --name=AccountingShare
```

deletes the existing volume with the name “AccountingShare”.

config volume list

This command lists all volumes that are configured on the array. The list includes the volume name, ID, the name of the storage pool the volume is configured on, the used and maximum size of the volume, and the date and time of the last snapshot created for the volume. This command also shows the protocol being used to share the volume. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
config volume list
```

lists all the volumes configured on the array.

config volume show

This command displays detailed information about the volume specified. The display includes the information listed in config volume list on page 73, and also shows the creation date, if compression is used, if the volume is configured to update time stamps when a file is read, and if the volume is configured as read only. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name of a specific volume for which you want to view detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config volume show --id=AccountingShare
```

displays detailed information about the volume named "AccountingShare".

config volume startmove

This command moves a volume from one storage pool to another. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the volume that you want to move to a different storage pool.
pool=<value>	The name of the storage pool to which you want to move the volume.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config volume startmove --ID=Volume1 --pool="Pool1"
```

starts moving a volume with the name of "Volume1" to the storage pool with the name "Pool1".

config volume update

This command updates an existing volume on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name of the volume you want to update.
atime=<true or **false>** (optional)	Whether the access time attribute for files is updated when they are read. If false, this avoids producing write traffic while reading files and can improve performance.
compression=<true or **false>** (optional)	Whether the array compresses data written to the volume. **Note:** Changing the compression setting only affects data written to the volume after the compression setting is changed. It does not impact data already written to the volume.
minsize=<value> (optional)	Capacity in bytes. You can enter values using KB, MB, GB, and TB. This space is allocated immediately if sufficient space is available. If insufficient space is available, then volume creation fails. If set to zero, no space is pre-allocated.
maxsize=<value> (optional)	Capacity in bytes. You can enter values using KB, MB, GB, and TB. If set to zero, no maximum size is configured. **Note:** Volumes are thin provisioned, so it is possible for the combined allocated maximum storage of all volumes to exceed the size of the storage pool.
readonly=<true or **false>** (optional)	Whether the volume is configured as read only and no data can be written to, or modified, on the volume.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
config volume update --id=AccountingShare
--compression=false --minsize=100GB --maxsize=0
```

updates the existing volume with the name "AccountingShare". This volume is updated to not use compression, and has a minimum size of 100 GB, with no maximum size.

CHAPTER 4

exit Command

This chapter provides instructions for using the `exit` command.

Note: At any time in the command line interface, you can type `help <command>` for detailed information about the command and any required parameters. You can also use the question mark character '?' in place of `help`.

exit

The `exit` command logs the current user out of the command line interface. It has no effect on the web interface. Use this command in conjunction with the following parameter.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

`exit`

logs the current user out of the command line interface.

CHAPTER 5

help Command

This chapter provides instructions for using the `help` command.

help

The `help` command displays help text about the specified command. There are no subcommands for this command. There are no parameters for this command.

Usage example

`help <command>`

displays help text for the command specified.

Note: The question mark character '?' can be used in place of `help`.

CHAPTER 6

history Command

This chapter provides instructions for using the `history` command.

history

The `history` command displays the last 100 commands run in the command line interface. This list does not persist through log outs. There are no subcommands for this command. There are no parameters for this command.

Usage example

`history`

displays the last 100 commands run in the command line interface.

CHAPTER 7

network Command

This chapter provides instructions for using the `network` command, and also lists and defines all subcommands of the `network` command.

network

The `network` command and its set of subcommands are used to configure the management, and data ports on the Verde array master node, as well as to configure DNS and SMTP settings.

All commands are listed in alphabetical order.

Note: At any time in the command line interface, you can type `help <command>` for detailed information about the command and any required parameters. You can also use the question mark character '?' in place of `help`.

network datainterface

Use the `network datainterface` commands to configure a single data port, as well as to display information on configured ports.

Note: By default, you can only create one data connection to the array. You can configure link aggregation of the same type of ports for better performance.

network datainterface list

This command lists all data interface ports on the array. This command shows the current configuration for each port, which includes link status, link speed, DHCP state, and IP address information. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
network datainterface list
```

displays a list of all data ports on the array.

network datainterface show

This command displays detailed information about the specified data port. The display includes the information displayed with network datainterface list, as well as additional information, such as the port MAC address, and MTU setting. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name or ID of the data port for which to display more detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network datainterface show --id='Data 1'
```

displays detailed information about the data port named "Data 1".

network datainterface update

This command configures the data ports on the master node. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name of the data interface you want to configure.
dhcp=<true or **false>** (optional)	Whether DHCP is enabled for the data interface and it automatically obtains an IP address from the DHCP server.
gateway=<value> (optional)	The gateway address for the data port.
ipaddress=<value> (optional)	The IP addressing for the data port. Enter a single IP address, or a comma separated list of addresses. **Note:** If the setting for DHCP is **false** you must specify an IP address.
mtu=<value> (optional)	The MTU (maximum transmission unit) setting, in bytes, for the data port. Enter a range between 1500-9000.
netmask=<value> (optional)	The netmask addressing for the data port. Enter a single netmask address, or a comma separated list of addresses. **Note:** If the setting for DHCP is **false** you must specify a netmask.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network datainterface update --id='Data 1' --dhcp=false
--ipaddress=10.10.10.100 --netmask=255.255.255.0
--gateway=10.10.10.1 --mtu=1500
```

configures the data port named "Data 1". This data port uses a static IP address with the information entered with the command. The data port has an MTU value of 1500.

network datalagg

Use the `network datalagg` commands to configure, delete, and update a data link aggregation on the array, as well as to view detailed information about the link aggregation.

Note: The array only supports one data connection. You can configure link aggregation of the same type of ports for better performance.

network datalagg create

This command configures a link aggregation on the array. Use this command with the following parameters. .

Parameter	Definition
dhcp=<true or **false>**	Whether DHCP is enabled for the data interface and it automatically obtains an IP address from the DHCP server.
mode=<value>	The type of ports to use to create the link aggregation. You can aggregate all ports of the same type and speed. **Values:** 1000Base-T, 10GigE
gateway=<value> (optional)	The gateway address for the data port.
ipaddress=<value> (optional)	The IP addressing for the data port. Enter a single IP address, or a comma separated list of addresses. **Note:** If the setting for DHCP is **false** you must specify an IP address.
netmask=<value> (optional)	The netmask addressing for the data port. Enter a single netmask address, or a comma separated list of addresses. **Note:** If the setting for DHCP is **false** you must specify a netmask.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network datalagg create --dhcp=false
--mode=10GigE --ipaddress=10.10.10.100
--netmask=255.255.255.0 --gateway=10.10.10.1
```

configures a link aggregation using all 10 GigE ports. This link aggregation uses a static IP address with the information entered with the command.

network datalagg delete

This command deletes an existing data link aggregation on the array. Use this command with the following parameters.

Parameter	Definition
id=<value>	The ID of the link aggregation you want to delete. Use network datalagg list to determine the ID of the link aggregation.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network datalagg delete --id=lagg0
```

deletes the link aggregation with the ID “lagg0”.

network datalagg list

This command shows the current link aggregation configuration, which includes a list of ports configured in link aggregation, link status, DHCP state, and IP address information. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
network datalagg list
```

displays the link aggregation configured on the array.

network datalagg show

This command displays detailed information about the link aggregation configured on the array. The display includes the information displayed with network datalagg list on page 84, as well as additional information, such as the port MAC address and MTU setting. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
network datalagg show
```

displays detailed information about the link aggregation configured on the array.

network datalagg update

This command updates an existing link aggregation. You cannot use this command to change a link aggregation from using all ports of one type to using all ports of another type. If you want to change an existing link aggregation to use different ports, you must delete the existing link aggregation (see network datalagg delete on page 84) and create a new one with the desired ports (see network datalagg create on page 83).

Use this command with at least one of the following parameters.

Parameter	Definition
dhcp=<true or **false>** (optional)	Whether DHCP is enabled for the data interface and it automatically obtains an IP address from the DHCP server.
gateway=<value> (optional)	The gateway address for the data port.
ipaddress=<value> (optional)	The IP addressing for the data port. Enter a single IP address, or a comma separated list of addresses. **Note:** If the setting for DHCP is **false** you must specify an IP address.
netmask=<value> (optional)	The netmask addressing for the data port. Enter a single netmask address, or a comma separated list of addresses. **Note:** If the setting for DHCP is **false** you must specify a netmask.
mtu=<value> (optional)	The MTU (maximum transmission unit) setting, in bytes, for the data port. Enter a range between 1500-9000.

Parameter	Definition
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network datalagg update --dhcp=true --mtu=9000
```

updates the current link aggregation to use DHCP to obtain an IP address, and sets the MTU value at 9000.

network dnssearchdomain

Use the `network dnssearchdomain` commands to create, delete, or view information about a DNS search domain. A DNS search domain defines a DNS suffix. The DNS suffix is added to the configured host name to create a fully qualified domain name.

network dnssearchdomain create

This command configures a DNS search domain on the array. Use the command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The DNS search domain.
insertasfirst (optional)	Adds the search domain to the top of the list, rather than the bottom. DNS search domains are used in the order specified. Use this parameter to configure the search domain to have top priority.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network dnssearchdomain create --id=sldomain.com
```

creates a DNS search domain with the address of “sldomain.com”.

network dnssearchdomain delete

This command deletes a previously configured DNS search domain. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The search domain you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network datalagg delete --id=sldomain.com
```

deletes the DNS search domain with the address of “sldomain.com”.

network dnssearchdomain list

This command lists all DNS search domains configured on the array in the order they are used. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
network dnssearchdomain list
```

lists all DNS search domains configured on the array.

network dnsserver

Use the `network dnsserver` commands to create, delete, and view the details of existing DNS server entries. A DNS server resolves domain names into IP addresses.

network dnsserver create

This command creates a DNS server entry on the array. Use this command with the following parameters.

Parameter	Definition
id=<value>	The IP address of the DNS server.
insertasfirst (optional)	Adds the DNS server entry to the top of the list, rather than the bottom. DNS servers are used in the order specified. Use this parameter to configure the server to have top priority.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network dnsserver create --id=10.10.10.100
--insertfirst
```

creates a DNS server with the address of 10.10.10.100. The new DNS server is added to the top of the list of servers.

network dnsserver delete

This command deletes an existing DNS server configured on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The IP address of the DNS server you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network dnsserver delete --id=10.10.10.100
```

deletes the DNS server with the specified IP address.

network dnsserver list

This command lists all DNS severs configured on the array in the order they are used. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
network dnsserver list
```

lists all DNS servers configured on the array.

network managementinterface

Use the `network management` interface commands to configure the array management port, and to view its current configuration.

network managementinterface show

This command displays the current configuration of the array management port, which includes link status, link speed, DHCP state, and IP address information. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
network managementinterface show
```

displays configuration information for the array management port.

network managementinterface update

This command configures the array management port.

Use this command with at least one of the following parameters.

Parameter	Definition
dhcp=<true or **false>** (optional)	Whether DHCP is enabled for the data interface and it automatically obtains an IP address from the DHCP server.
gateway=<value> (optional)	The gateway address for the management port.

Parameter	Definition
ipaddress=<value> (optional)	The IP address for the management port. **Note:** If the setting for DHCP is **false** you must specify an IP address.
mtu=<value> (optional)	The MTU (maximum transmission unit) setting, in bytes, for the data port. Enter a range between 1500-9000.
netmask=<value> (optional)	The netmask address for the management port. **Note:** If the setting for DHCP is **false** you must specify a netmask.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network managementinterface update --dhcp=false
--ipaddress=10.10.10.100 --netmask=255.255.255.0
--gateway=10.10.10.1 --mtu=1500
```

configures the array management port to use a static IP address with the information entered with the command. The port has an MTU value of 1500.

network nslookup

The `network nslookup` command is used to look up a DNS server. This command has no subcommands. The command displays the name of the DNS server, the server IP address, and the fully qualified domain name. This command has no subcommands. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The DNS server name or IP address to look up.
managementinterface (optional)	Uses the management port to look up the name of the DNS server. ▪ If omitted, a data port connection is used to look up the DNS server.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network nslookup --id=10.0.1.9
```

looks up the specified DNS sever and displays information about the server.

network ping

Use the `network ping` command to ping a remote connection from the master node. This command has no subcommands. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The DNS name or IP address to ping.
managementinterface (optional)	Uses the management port to ping a remote connection. ▪ If omitted, a data port connection is used to ping the remote connection.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network ping --id=10.10.10.100 --managementinterface
```

pings the remote connection with an IP address of 10.10.10.100. The ping is performed using the management port.

network smtp

Use the `network smtp` commands to configure the array to send emails to a specified SMTP server, as well as to view the current configuration.

network smtp show

This command displays the current SMTP configuration on the array. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
network smtp show
```

displays the current SMTP configuration on the array.

network smtp update

This command configures the array to send emails to the specified SMTP server. Use this command in conjunction with at least one of the following parameters.

Parameter	Definition
fromaddress=<value> (optional)	The email address that is used as the 'from' address for all emails generated by the array.
port=<value> (optional)	The TCP port number of the SMTP server.
server=<value> (optional)	The IP address or DNS name for the SMTP server.
tls=<true or **false>** (optional)	Whether TLS (Transport Layer Security) credentials are used when sending emails to the SMTP server.
uauser=<value> (optional)	The user name for TLS authentication on the SMTP server. ▪ If **tls** is set to **true** and this parameter is omitted, the interface prompts you for this information after you run the command.
uapassword=<value> (optional)	The password for TLS authentication on the SMTP server. ▪ If **tls** is set to **true** and this parameter is omitted, the interface prompts you for this information after you run the command. **Note:** Providing information via command parameters is less secure than via a prompt.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
network smtp update
--fromaddress='Verde1@yourcompany.com'
--port=162 --server=10.10.10.100 --tls=false
```

configures the array to send emails to a sever with an IP address of 10.10.10.100 using port number 162. The SMTP server does not require TLS authentication.

network summary

The network summary command displays a summary of the network configuration on the array. The command also displays configured DNS servers, DNS search domains, and SMTP configuration. This command has no subcommands. There are no parameters for this command.

Usage example

```
network summary
```

displays a summary of the current network, DNS, and SMTP configuration on the array.

CHAPTER 8

power Command

This chapter provides instructions for using the `power` command, and also lists and defines all subcommands of the `power` command.

power

The `power` command and its set of subcommands are used to power off, or reboot the master node.

All commands are listed in alphabetical order.

Note: At any time in the command line interface, you can type `help <command>` for detailed information about the command and any required parameters. You can also use the question mark character '?' in place of `help`.

power reboot

Use the `power reboot` command to reboot the master node. This command has no subcommands. Use this command in conjunction with the following parameters.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.

Usage example

```
power reboot
```

reboots the master node. You are asked to confirm this command.

power shutdown

Use the `power shutdown` command to power off the master node. This command has no subcommands. Use this command in conjunction with at least one of the following parameters.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.

Usage example

```
power shutdown
```

powers off the array. You are asked to confirm this command.

CHAPTER 9

status Command

This chapter provides instructions for using the `status` command, and also lists and defines all subcommands of the `status` command.

status

The `status` command and its set of subcommands are used to view the current status of the array, as well as the status of its components.

All commands are listed in alphabetical order.

Note: At any time in the command line interface, you can type `help <command>` for detailed information about the command and any required parameters. You can also use the question mark character '?' in place of `help`.

status bootdrive

Use the `status bootdrive` command to list the boot drives installed in the master node, and view detailed information about a specified boot drive.

status bootdrive list

This command lists all boot drives installed in the master node. The display shows the drive number, size, and status. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status bootdrive list
```

displays a list of the boot drives installed in the master node.

status bootdrive show

This command shows detailed information about the boot drive specified. This command displays the same information as status bootdrive list, as well as displaying the boot drive manufacturer, model, and serial number. Use the command with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The ID of the boot drive for which to display more detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status bootdrive show --id=1
```

displays detailed information about the boot drive with the ID “1”.

status chassis

Use the `status chassis` commands to display a list of all chassis, to display detailed information about a specified chassis, and to turn on and off the beacon feature. When turned on, the beacon blinks the Visual Status Beacon LED in the front bezel, so it is easy to find in your environment. This can be useful if you have multiple arrays and need to replace a drive in one of them.

status chassis list

This command displays a list of all of the chassis that comprise the array. This display includes the master node and any expansion nodes attached to the master node. The display includes the status of each chassis, its ID and serial number, the number of drives installed, and the capacity of the chassis. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status chassis list
```

displays a list of all chassis that comprise the array.

status chassis show

This command displays detailed information about a specified chassis in the array. This command displays the same information as status chassis list, as well as displaying the node type. Use this command with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The ID or serial number of the chassis for which to display more detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status chassis show --id=500304815162342
```

displays detailed information about the chassis with a serial number of "500304815162342".

status chassis update

This command turns the beacon feature on and off. The beacon does not turn off automatically and must be manually turned off. Use this command with the following parameters.

Note: Not all chassis include an active bezel with Visual Status Beacon.

Parameter	Definition
id=<value>	The ID or serial number of the chassis.
beacon=<true or **false>** (optional)	Whether the beacon feature is activated and the Visual Status Beacon LED flashes blue. If false, the LED stops flashing.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status chassis show --id=500304815162342 --beacon=true
```

turns on the beacon for the chassis with a serial number of "500304815162342".

status cpu

Use the `status cpu` commands to display a list of all CPUs present in the master node, as well as viewing detailed information about a specified CPU.

status cpu list

This command lists all CPUs present in the master node. The display shows the CPU slot number, status, and current temperature. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status cpu list
```

displays a list of all CPUs in the master node.

status cpu show

This command displays detailed information about a specified CPU. This command displays the same information as status cpu list on page 99, as well as displaying the CPU shutdown temperature. Use this command with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The slot number of the CPU for which to display more detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status cpu show --id=1
```

displays detailed information about the CPU with the slot number "1".

status drive

Use the `status drive` commands to display a list of all data drives installed in the array, as well as displaying detailed information for a specified drive.

status drive list

This command lists all data drives in the array. The information displayed includes the drive slot number, drive status, interface type, capacity, the system name of the array where the drive is installed, the pool name to which the drive is assigned, and the drive ID number. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status drive list
```

displays a list of all data drives installed in the array.

status drive show

This command displays detailed information for a specified data drive. This command displays the same information as status drive list on page 100, as well as displaying the drive manufacturer, firmware version, model and serial number, and speed. Use this command with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The ID of the data drive for which you want to display more detailed information, in single quotes. **Note:** Drive ID numbers are very long. It may be easier to copy the drive ID from status drive list on page 100and paste it into this command.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status drive show --id='id1,enc@500304815162342/type0/
slot@6/elmdesc@Slot 6'
```

displays detailed information about the data drive with the ID specified in the command.

status fan

Use the `status fan` commands to display a list of all fans in the array, as well as displaying detailed information about a specified fan.

status fan list

This command lists all fans installed in the array. The display includes the fan slot number, status, RPM speed, and ID number. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status fan list
```

displays a list of all fans installed in the array.

status fan show

This command displays detailed information about a specified fan. This command displays the same information as status fan list on page 101. Use this command with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The ID of the fan for which you want to display more detailed information, in single quotes. **Note:** Some fan ID numbers are very long. It may be easier to copy the fan ID from status fan list on page 101 and paste it into this command.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status fan show --id='id1,enc@500304815162342/type@2/
slot@2/elmdesc@Fan2'
```

displays detailed information about the fan with the ID specified in the command.

status message

Use the `status message` command to list all messages that are present on the array.

status message list

This command lists all messages on the array. This command has no parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status message list
```

displays a list of all messages on the array.

status performance

Use the `status performance` commands to view the performance of various components of the array, including the CPUs, drives, network, and storage pools.

status performance cpu list

This command lists all CPUs installed in the array and displays each CPU's utilization and idle percentages. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status performance cpu list
```

displays a list of all CPUs installed in the array and the performance for each CPU.

status performance cpu show

This command displays performance data for a specified CPU. This command displays the same information as status performance cpu list. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The ID of the CPU for which you want to display performance data.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status performance cpu show --id=1
```

displays performance data about the CPU with the ID "1".

status performance drive list

This command lists all data drives installed in the array and displays each drive's read and write speeds, and the read and write latency. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status performance drive list
```

displays a list of all drives installed in the array and the performance for each drive.

status performance drive show

This command displays performance data for a specified data drive. This command displays the same information as status performance drive list. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name, ID, or slot number of the drive for which you want to display performance data.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status performance drive show --id=da8
```

displays performance data about the data drive with the ID "da8".

status performance network list

This command lists all network ports in the array and displays each port's read and write speeds, and the number of packets read and written. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status performance network list
```

displays a list of all network ports in the array and the performance for each network port.

status performance network show

This command displays performance data for a specified network port. This command displays the same information as status performance network list. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name or ID of the network port for which you want to display performance data.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status performance network show --id=igb2
```

displays performance data about the network port with the ID "igb2".

status performance pool list

This command lists all storage pools configured on the array and displays each pool's read and write speeds, and the number of read and write operations. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status performance pool list
```

displays a list of all storage pools configured on the array and the performance for each pool.

status performance pool show

This command displays performance data for a specified storage pool. This command displays the same information as status performance pool list. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name or ID of the storage pool for which you want to display performance data.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status performance pool show --id=Pool1
```

displays performance data about the storage pool with the ID "Pool1".

status powersupply

Use the `status powersupply` commands to display a list of all power supplies installed in the array, as well as displaying detailed information about a specified power supply.

status powersupply list

This command lists all power supplies installed in the array. The display includes the power supply status, slot number, system name, ID number, and current wattage. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status powersupply list
```

displays a list of all power supplies installed in the array.

status powersupply show

This command displays detailed information about a specified power supply. This command displays the same information as status powersupply list. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The ID of the power supply for which you want to display more detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
status powersupply show --id=1
```

displays detailed information about the power supply with the ID “1”.

status summary

Use the `status summary` command to see an overview of the array. The display includes a list of all drives installed in the array, the hardware status, storage pool information, and the network configuration. There are no subcommands for this command.

Usage example

```
status summary
```

displays a summary of the status of the array.

status system

Use the status system command to view basic information about the master node. The information includes the system name, serial number, software version, part number, system memory, and if safe mode is enabled.

status system show

This command displays basic information about the array. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
status system show
```

displays basic information about the master node.

CHAPTER 10

support Command

This chapter provides instructions for using the `support` command, and also lists and defines all subcommands of the `support` command.

support

The `support` command and its set of subcommands are used to generate log sets, enter contact information and activation keys, and check for any cabling errors.

All commands are listed in alphabetical order.

Note: At any time in the command line interface, you can type `help <command>` for detailed information about the command and any required parameters. You can also use the question mark character '`?`' in place of `help`.

support activationkey

Use the `support activationkey` commands to enter, and delete action keys, as well as to display the current activation keys on the array.

support activationkey create

This command enters an activation key on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
key=<value>	The key provided by Spectra Logic in single quotes.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support activationkey create --key='4815162342'
```

enters a key on the array with the value specified in the command.

support activationkey delete

This command deletes an activation key on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The ID of the activation key you want to delete in quotes.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support activationkey delete --id=1
```

deletes the activation key on the array with the ID of '1'.

support activationkey list

This command lists all activation keys entered on the array. The display shows the key ID, the key itself, and the key expiration date. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
support activationkey list
```

lists all activation keys on the array.

support activationkey show

Use this command to show detailed information about the specified activation key. This command displays the same information as support activationkey list. Use this command with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The ID of the activation key for which you want to view detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support activationkey show --id=1
```

displays detailed information about the activation key on the array with the ID of '1'.

support customerinfo

Use the `support customerinfo` commands to configure the customer information on the array. Customer information includes contact information, location of the array, and a space for any additional information. You can also display the existing customer information on the array.

support customerinfo show

This command displays the existing customer information that is entered on the array. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
support customerinfo show
```

displays the existing customer information on the array.

support customerinfo update

This command configures the customer information on the array. Use this command in conjunction with at least one of the following parameters.

Parameter	Definition
firstname=<value> (optional)	The first name of the array administrator.
lastname=<value> (optional)	The last name of the array administrator.
companyname=<value> (optional)	Your company name.
address1=<value> (optional)	The address of the building where your array is located.
address2=<value> (optional)	Additional address space.
primaryphonenumber=<value> (optional)	The primary contact phone number for the array administrator.
alternatephonenumber=<value> (optional)	The alternate contact phone number for the array administrator.
emailaddress=<value> (optional)	The email address for the array administrator.
location=<value> (optional)	The location inside the building where your array is located.
contractnumber=<value> (optional)	The license number of your support contract.
notes=<value> (optional)	Space for any notes you want to include in your customer information.

Parameter	Definition
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support customerinfo update --fistname=Desmond
--lastname=Hume --companyname='Dharma Initiative'
--address1='4 Swan Station Way'
--primaryphonenumber=481-516-2342
--emailaddress=desmond.hume@swanstation.di
--location='computer room'
```

configures the contact information on the array with the information specified in the command.

support internalsupportinfo

Use the `support internalsupportinfo` command to display detailed information about the array that Spectra Logic Technical Support uses during troubleshooting. Information displayed includes the current software version, BIOS information, and information about the HBAs installed in the array, including firmware, and driver versions.

support internalsupportinfo list

This command displays the internal support information for the array. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
support internalsupportinfo list
```

displays the internal support information for the array.

support kernellogset

Use the `support kernellogset` commands to list all kernel log sets on the array, view detailed information about a specified log set, and delete a log set. The array automatically generates kernel log sets when a kernel panic occurs. If a kernel panic occurs and the array generates a kernel log set, contact Spectra Logic Technical Support (see Contacting Spectra Logic on page 4).

support kernellogset delete

This command deletes an existing kernel log set on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name of the log set you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support kernellogset delete --id=kernellogset_070713
```

deletes the kernel log set with the name “kernellogset_070713”.

support kernellogset list

This command lists all kernel log sets on the array. The information displayed includes the ID number of each kernel log set, the creation date, and the size of the log set. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
support kernellogset list
```

displays the kernel log sets present on the array.

support kernellogset show

This command displays detailed information about a specified kernel log set. This command displays the same information as support kernellogset list on page 114. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name of the kernel log set for which you want to view detailed information.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support kernellogset show --id=kernellogset_070713
```

displays detailed information about the kernel log set with the name "kernellogset_070713".

support logset

Use the `support logset` commands to create, delete, and email log sets, as well as to view all log sets present on the array, and to view detailed information about a specified log set. Log sets contain information about the state of the array when the array genreated the log set and can be useful in troubleshooting.

support logset create

This command creates a log set on the array. There are no parameters for this command.

Usage example

```
support logset create
```

creates a log set on the array.

support logset delete

This command deletes an existing log set on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name of the log set you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support logset delete --id=mannual_2013-07-07_15-31-08
```

deletes the log set with the name "manual_2013-07-07_15-31-08".

support logset email

This command emails an existing log set to a previously configured mail recipient on the array. You cannot email a log set directly to an email address, you must first create a mail recipient with that address. To create a mail recipient, see config mailrecipient on page 30. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name or ID of the log set you want to email.
mailrecipient=<value>	The name or ID of the mail recipient you want to receive the log set.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support logset email --id=mannual_2013-07-07_15-31-08
--mailrecipient=1
```

emails the log set with the name "manual_2013-07-07_15-31-08" to the mail recipient with the ID of "1".

support logset list

This command lists all log sets present on the array. There are no parameters for this command. The display includes the log set name, creation date, and size.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
support logset list
```

lists all log sets that are present on the array.

support logset show

This command displays detailed information for the log set specified. This command displays the same information as support logset list. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name of the log set for which you want to view detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support logset show --id=mannual_2013-07-07_15-31-08
```

displays detailed information for the log set with the name “manual_2013-07-07_15-31-08”.

support logset schedule

Use the `support logset schedule` commands to create, delete, and view all log set schedules present on the array, and to view detailed information about a specified log set schedule. Log sets schedules determine when the array generates a log set.

support logset schedule create

This command creates a log set schedule on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
daysofweek=<value>	A comma-separated list of the days of the week that a log set is generated.
starthour=<value>	The hour of the day that the schedule generates a log set.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support logset schedule create
--daysofweek='monday,wednesday' --starthour=13
```

creates a log set schedule on the array. The schedule generates a log set every Monday and Wednesday at 1 pm.

support logsetschedule delete

This command deletes an existing log set schedule on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name of the log set schedule you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support logset schedule delete --id=2013-10-13
```

deletes a log set schedule with the name "2013-10-13".

support logsetschedule list

This command lists all log set schedules that are configured on the array. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
support logsetschedule list
```

lists all log sets schedules that are present on the array.

support logsetschedule show

This command displays detailed information for the log set schedule specified. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name of the log set schedule for which you want to view detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support logsetschedule show --id=stat_2013-10-13
```

displays detailed information for the log set schedule with the name “2013-10-13”.

support statisticslogset

Use the `support statisticslogset` commands to create and delete, statistic log sets, as well as to view all statistic log sets present on the array, and to view detailed information about a specified statistic log set. The statistics log sets are not human readable and contain information about the performance of the array when the array generated the log set, which can be useful in troubleshooting.

support statisticslogset create

This command creates a statistic log set on the array. There are no parameters for this command.

Usage example

```
support statisticslogset create
```

creates a statistic log set on the array.

support statisticslogset delete

This command deletes an existing statistic log set on the array. Use this command in conjunction with the following parameters.

Parameter	Definition
id=<value>	The name of the statistic log set you want to delete.
force (optional)	Forces the command to execute, even if the command is unsafe. This also disables any required user confirmations.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support statisticslogset delete
--id=stat_2013-07-08_14-45-01
```

deletes the statistics log set with the name “stat_2013-07-08_14-45-01”.

support statisticslogset list

This command lists all statistic log sets present on the array. The display includes the log set name, creation date, and size. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
support statisticslogset list
```

lists all statistics log sets that are present on the array.

support statisticslogset show

This command displays detailed information for the statistic log set specified. This command displays the same information as support statisticslogset list on page 120. Use this command in conjunction with the following parameters.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Parameter	Definition
id=<value>	The name of the statistic log set for which you want to view detailed information.
noprogress (optional)	Turns off the progress spinner that indicates that the CLI is waiting for a response from a lower-level component. The progress spinner could interfere with a script's ability to parse a command's output.

Usage example

```
support statisticslogset show
--id=stat_2013-07-08_14-45-01
```

displays detailed information for the log set with the name "stat_2013-07-08_14-45-01".

support supportinfo

Use the `support supportinfo` command to display the contact information for Spectra Logic, Spectra Logic Technical Support, and Spectra Logic sales.

support supportinfo show

This command displays the contact information for Spectra Logic. There are no parameters for this command.

Note: This command accepts using scripting parameters. See Scripting on page 17 for more information.

Usage example

```
support supportinfo show
```

displays the contact information for Spectra Logic.