NetApp

Command formatting

SANtricity commands

NetApp
May 25, 2022

Table of Contents

Command formatting

Overview of CLI

The command line interface (CLI) is a software application that provides a way to configure and monitor storage arrays. Using the CLI, you can run commands from an operating system prompt, such as the DOS `C:` prompt, a Linux operating system path, or a Solaris operating system path.

The CLI gives you direct access to a script engine that is a utility in the storage management software. The script engine runs commands that configure and manage the storage arrays. The script engine reads the commands, either through a script file or directly from the command line, and performs the operations instructed by the commands.

The script commands configure and manage a storage array. The script commands are distinct from the CLI commands. You can enter individual script commands, or you can run a file of script commands. When you enter an individual script command, you embed the script command in a CLI command. When you run a file of script commands, you embed the file name in the CLI command.

Some software features described in this document might not be available for your E-Series Storage System. For questions about available features, contact your account representative.

The CLI is a feature of the SANtricity Storage Manager software.

CLI structure

Structure of CLI commands

The CLI commands are in the form of a command wrapper and elements embedded into the wrapper. A CLI command consists of these elements:

- A command wrapper identified by the term `SMcli`
- The storage array identifier
- Terminals that define the operation to be performed
- Script commands

The CLI command wrapper is a shell that identifies storage array controllers, embeds operational terminals, embeds script commands, and passes these values to the script engine.

All CLI commands have the following structure:

```
SMcli *storageArray terminal script-commands*;
```

- `SMcli` invokes the command line interface.
- `storageArray` is the name or the IP address of the storage array.
- `terminal` is a CLI value that defines the environment and the purpose for the command.
- `script-commands` are one or more script commands or the name of a script file that contains script commands. (The script commands configure and manage the storage array.)

If you enter an incomplete or inaccurate `SMcli` string that does not have the correct syntax, parameter names, options, or terminals, the script engine returns usage information.

Interactive mode

If you enter `SMcli` and a storage array name, but do not specify CLI parameters, script commands, or a script file, the command line interface runs in interactive mode. Interactive mode lets you run individual commands without prefixing the commands with `SMcli`.

In interactive mode, you can enter a single command, view the results, and enter the next command without typing the complete `SMcli` string. Interactive mode is useful for determining configuration errors and quickly testing configuration changes.

To end an interactive mode session, type the operating system-specific command. For Linux, this key combination is **Control-D**. For Windows, this key combination is **Control-Z + ENTER**.

CLI command wrapper syntax

General syntax forms of the CLI command wrappers are listed in this section. The conventions used in the CLI command wrapper syntax are listed in the following table.

Convention	Definition
`a | b`	Alternative ("a" or "b")
italicized-words	Needs user input to fulfill a parameter (a response to a variable)
`[ ... ]` (square brackets)	Zero or one occurrence (square brackets are also used as a delimiter for some command parameters)
`{ ... }` (curly braces)	Zero or more occurrences
`(a | b | c)`	Choose only one of the alternatives
`a &| b`	And/or. This is used for https client mode, when you can use with one or both controller IP addresses. This way, if one controller is not responding, SMcli will use the alternative IP address. This also covers the case when both IP addresses are required, such as for firmware download.

ⓘ To run all of the CLI commands you must have administrator privileges. Some CLI commands will run without administrator privileges. Many commands, however, will not run. If the CLI command does not run because you do not have correct privileges, the CLI returns an exit code of 12.

Example in https client mode

The following examples demonstrate the `https` client mode command line parameters described in Command line parameters.

```
SMcli (Controller A host-name-or-IP-address&|
Controller B host-name-or-IP-address)  -u username -p password -c
"commands;" [-clientType (auto | https | symbol)]
```

> (i) If you do not specify a `clientType`, but do include the `-u` option and the `username` variable, the system will use either `https` or `symbol` client mode, whichever is available.

Examples in symbol client mode

The following examples demonstrate the `symbol` client mode command line parameters described in Command line parameters.

```
SMcli **-a** **email:** email-address [host-name-or-IP-address1 [host-
name-or-IP-address2]] [**-n** storage-system-name | **-w** wwID | **-h**
host-name] [**-I** information-to-include] [**-q** frequency] [**-S**]
```

> (i) The `-a` command line option is not supported for the E2800 or E5700 storage array.

```
SMcli **-x** **email:** email-address [host-name-or-IP-address1 [host-
name-or-IP-address2]] [**-n** storage-system-name | **-w** wwID | **-h**
host-name] [**-S**]
```

> (i) The `-x` command line option is not supported for the E2800 or E5700 storage array.

```
SMcli (**-a** | **-x**) **trap:** community, host-name-or-IP-address
[host-name-or-IP-address1 [host-name-or-IP-address2]] [**-n** storage-
system-name | **-w** wwID | **-h** host-name] [**-S**]
```

> (i) The `-a` and `-x` command line options are not supported for the E2800 or E5700 storage array.

```
SMcli **-d** [**-w**] [**-i**] [**-s**] [**-v**] [**-S**]
```

> (i) The `-s` command line option is not supported for the E2800 or E5700 storage array.

```
SMcli host-name-or-IP-address **-F** email-address [**-g**
contactInfoFile] [**-S**]
```

```
SMcli **-A** [host-name-or-IP-address [host-name-or-IP-address]] [**-S**]
```

```
SMcli **-X **(**-n** storage-system-name | **-w** wwID | **-h** host-name)
```

```
SMcli **-?**
```

Command line parameters

New command line parameters in the 11.40 release

The SANtricity OS 11.40 release introduces, for the E2800 and E5700 controllers with embedded web services, the ability to interact on the Command Line using a secure HTTPS protocol. These controllers can optionally use the SYMbol protocol for Command Line interactions instead. The SYMbol protocol is the only supported protocol for the E2700 and E5600 controllers. To preserve existing scripts and minimize transition time, the CLI options and grammar are preserved as much as possible. However, there are some differences in the capabilities of the E2800 and E5700 controllers with regards to security, authentication, AutoSupport, and alert messaging that render some of the CLI grammar for those controllers obsolete. However, in some cases the grammar is only obsolete on the E2800 or E5700 when the new https protocol is used.

For the new parameters that only apply to the `https` client type, it follows that they also apply only to the E2800 or E5700 controllers.

For a list of new, deprecated, and obsolete commands with this release, see Changes in the CLI and Script commands in this release.

Parameter	Definition
`-clientType`	This argument forces the creation of an appropriate script engine. Use this optional parameter with one of the following values: • `auto` - Device discovery is automatically performed to detect the appropriate script engine type. • `https` - A REST-based script engine is created. • `symbol` - A SYMbol-based script engine is created.

Parameter	Definition
-u	Follow this parameter with the *username* variable. The username is only required for the **https** client type. This argument is not applicable to the **symbol** client type and will be silently ignored. If the username argument is specified, device discovery is performed to determine the correct client type (**https** vs. **symbol**).
-P	This argument requires one of two arguments: • *password_file* - Contains the password to use for authentication. • – (one dash) - Read the password from `stdin`. Note that the addition of this argument is applicable to all controllers, regardless of whether the **https** client type or **symbol** client type is used.
-k	This optional argument allows an **https** client to operate in insecure mode. This means that the storage array's certificate will not be validated. By default, if omitted, the proper authentication will be performed. This argument is not applicable to the **symbol** client type and will be silently ignored.

Command line parameters that only apply to E2700 or E5600 controllers

Because the E2700 and E5600 controllers do not have embedded alert management capabilities, these command line parameters are applicable. These parameters are not applicable to the E2800 or E5700 controllers.

Parameter	Definition
`-a`	Adds a Simple Network Management Protocol (SNMP) trap destination or an email address alert destination. • When you add an SNMP trap destination, the SNMP community is automatically defined as the community name for the trap, and the **host** is the IP address or Domain Name Server (DNS) host name of the system to which the trap should be sent. • When you add an email address for an alert destination, the **email-address** is the email address to which you want the alert message to be sent. ⓘ This command line option is obsolete for the E2800 and E5700 storage arrays. Use the RESTful API, SANtricity System Manager, or cURL commands.
`-m`	Specifies the host name or the IP address of the email server from which email alert notifications are sent. ⓘ This command line option is obsolete for the E2800 and E5700 storage arrays. Use the RESTful API, SANtricity System Manager, or cURL commands.
`-s` (lowercase)	Shows the alert settings in the configuration file when used with the `-d` parameter. ⓘ This command line option is obsolete for the E2800 and E5700 storage arrays. Use the RESTful API, SANtricity System Manager, or cURL commands.

Parameter	Definition
`-x` (lowercase)	Removes an SNMP trap destination or an email address alert destination. The `community` is the SNMP community name for the trap, and the `host` is the IP address or DNS host name of the system to which you want the trap sent. ⓘ This command line option is obsolete for the E2800 and E5700 storage arrays. Use the RESTful API, SANtricity System Manager, or cURL commands.

Command line parameters that apply to all controllers running with a symbol client type

Parameter	Definition
`-R` (uppercase)	Defines the user role for the password. The roles can be either: • **admin** — The user has privilege to change the storage array configuration. • **monitor** — The user has privilege to view the storage array configuration, but cannot make changes. The **-R** parameter is valid only when used with the **-p** parameter, which specifies that you define a password for a storage array. The **-R** parameter is required only if the dual password feature is enabled on the storage array. The **-R** parameter is not necessary under these conditions: • The dual password feature is not enabled on the storage array. • Only one admin role is set and the monitor role is not set for the storage array.

Command line parameters applicable to all controllers and all client types

Parameter	Definition
host-name-or-IP-address	Specifies either the host name or the Internet Protocol (IP) address (*xxx.xxx.xxx.xxx*) of an in-band managed storage array or an out-of-band managed storage array. • If you are managing a storage array by using a host through in-band storage management, you must use the `-n` parameter or the `-w` parameter if more than one storage array is connected to the host. • If you are managing a storage array by using out-of-band storage management through the Ethernet connection on each controller, you must specify the *host-name-or-IP-address* of the controllers. • If you have previously configured a storage array in the Enterprise Management Window, you can specify the storage array by its user-supplied name by using the `-n` parameter. • If you have previously configured a storage array in the Enterprise Management Window, you can specify the storage array by its World Wide Identifier (WWID) by using the `-w` parameter.
`-A`	Adds a storage array to the configuration file. If you do not follow the `-A` parameter with a *host-name-or-IP-address*, auto-discovery scans the local subnet for storage arrays.
`-c`	Indicates that you are entering one or more script commands to run on the specified storage array. End each command with a semicolon (`;`). You cannot place more than one `-c` parameter on the same command line. You can include more than one script command after the `-c` parameter.
`-d`	Shows the contents of the script configuration file. The file content has this format: *storage-system-name host-name1 host-name2*
`-e`	Runs the commands without performing a syntax check first.
`-F` (uppercase)	Specifies the email address from which all alerts will be sent.

Parameter	Definition
`-f` (lowercase)	Specifies a file name that contains script commands that you want to run on the specified storage array. The `-f` parameter is similar to the `-c` parameter in that both parameters are intended for running script commands. The `-c` parameter runs individual script commands. The `-f` parameter runs a file of script commands. By default, any errors that are encountered when running the script commands in a file are ignored, and the file continues to run. To override this behavior, use the `set session errorAction=stop` command in the script file.
`-g`	Specifies an ASCII file that contains email sender contact information that will be included in all email alert notifications. The CLI assumes that the ASCII file is text only, without delimiters or any expected format. Do not use the `-g` parameter if a `userdata.txt` file exists.
`-h`	Specifies the host name that is running the SNMP agent to which the storage array is connected. Use the `-h` parameter with these parameters: • `-a` • `-x`
`-I` (uppercase)	Specifies the type of information to be included in the email alert notifications. You can select these values: • `eventOnly` — Only the event information is included in the email. • `profile` — The event and array profile information is included in the email. You can specify the frequency for the email deliveries using the `-q` parameter.
`-i` (lowercase)	Shows the IP address of the known storage arrays. Use the `-i` parameter with the `-d` parameter. The file content has this format: *storage-system-name IP-address1 IPaddress2*

Parameter	Definition
`-n`	Specifies the name of the storage array on which you want to run the script commands. This name is optional when you use a *host-name-or-IP-address*. If you are using the in-band method for managing the storage array, you must use the `-n` parameter if more than one storage array is connected to the host at the specified address. The storage array name is required when the *host-name-or-IP-address* is not used. The name of the storage array that is configured for use in the Enterprise Management Window (that is, the name is listed in the configuration file) must not be a duplicate name of any other configured storage array.

`-o`	Specifies a file name for all output text that is a result of running the script commands. Use the `-o` parameter with these parameters:
	
	    • `-c`
	
	    • `-f`
	
	If you do not specify an output file, the output text goes to standard output (stdout). All output from commands that are not script commands is sent to stdout, regardless of whether this parameter is set.

`-p`	Defines the password for the storage array on which you want to run commands. A password is not necessary under these conditions: • A password has not been set on the storage array. • The password is specified in a script file that you are running. • You specify the password by using the `-c` parameter and this command: <code>set session password=password</code>

Parameter	Definition
`-P`	This argument requires one of two arguments: • *password_file* - contains the password to use for authentication. • `-`(dash) - read the password from `stdin`. Note that the addition of this argument is applicable to all controllers, regardless of whether the **https** client type or **symbol** client type is used.
`-q`	Specifies the frequency that you want to receive event notifications and the type of information returned in the event notifications. An email alert notification containing at least the basic event information is always generated for every critical event. These values are valid for the `-q` parameter: • `everyEvent` — Information is returned with every email alert notification. • `2` — Information is returned no more than once every two hours. • `4` — Information is returned no more than once every four hours. • `8` — Information is returned no more than once every eight hours. • `12` — Information is returned no more than once every 12 hours. • `24` — Information is returned no more than once every 24 hours. Using the `-I` parameter you can specify the type of information in the email alert notifications. • If you set the `-I` parameter to `eventOnly`, the only valid value for the `-q` parameter is `everyEvent`. • If you set the `-I` parameter to either the `profile` value or the `supportBundle` value, this information is included with the emails with the frequency specified by the `-q` parameter.

Parameter	Definition
`-quick`	Reduces the amount of time that is required to run a single-line operation. An example of a single-line operation is the `recreate snapshot volume` command. This parameter reduces time by not running background processes for the duration of the command. Do not use this parameter for operations that involve more than one single-line operation. Extensive use of this command can overrun the controller with more commands than the controller can process, which causes operational failure. Also, status updates and configuration updates that are collected usually from background processes will not be available to the CLI. This parameter causes operations that depend on background information to fail.
`-S` (uppercase)	Suppresses informational messages describing the command progress that appear when you run script commands. (Suppressing informational messages is also called silent mode.) This parameter suppresses these messages: • `Performing syntax check` • `Syntax check complete` • `Executing script` • `Script execution complete` • `SMcli completed successfully`
`-v`	Shows the current global status of the known devices in a configuration file when used with the `-d` parameter.
`-w`	Specifies the WWID of the storage array. This parameter is an alternate to the `-n` parameter. Use the `-w` parameter with the `-d` parameter to show the WWIDs of the known storage arrays. The file content has this format: *storage-system-name world-wide-ID IP-address1 IP-address2*
`-X` (uppercase)	Deletes a storage array from a configuration.
`-?`	Shows usage information about the CLI commands.

Script command structure

Structure of a script command

All script commands have the following structure:

```
*command operand-data* (*statement-data*)
```

- `command` identifies the action to be performed.
- `operand-data` represents the objects associated with a storage array that you want to configure or manage.
- `statement-data` provides the information needed to perform the command.

The syntax for *operand-data* has the following structure:

```
(*object-type* | all *object-types* | [*qualifier*] (*object-type*
[*identifier*] (*object-type* [*identifier*] | *object-types*
[*identifier-list*])))
```

An object can be identified in four ways:

- Object type — Use when the command is not referencing a specific object.
- `all` parameter prefix — Use when the command is referencing all of the objects of the specified type in the storage array (for example, `allVolumes`).
- Square brackets — Use when performing a command on a specific object to identify the object (for example, `volume [engineering]`).
- A list of identifiers — Use to specify a subset of objects. Enclose the object identifiers in square brackets (for example, `volumes [sales engineering marketing]`).

A qualifier is required if you want to include additional information to describe the objects.

The object type and the identifiers that are associated with each object type are listed in this table.

Object Type	Identifier
`controller`	a or b
`drive`	For enclosures with drawers, use tray ID, drawer ID, and slot ID. Alternatively, just tray ID and slot ID.
`replacementDrive`	For enclosures with drawers, use tray ID, drawer ID, and slot ID. Alternatively, just tray ID and slot ID.
`driveChannel`	Drive channel identifier

Object Type	Identifier
`host`	User label
`hostChannel`	Host channel identifier
`hostGroup`	User label
`hostPort`	User label
`iscsiInitiator`	User label or iSCSI Qualified Name (IQN)
`iscsiTarget`	User label or IQN
`storageArray`	Not applicable
`tray`	Tray ID
`volume`	Volume user label or volume World Wide Identifier (WWID) (`set` command only)
`volumeCopy`	Target volume user label and, optionally, the source volume user label
`volumeGroup`	User label Valid characters are alphanumeric, a hyphen, and an underscore.

Statement data is in the form of:

- Parameter = value (such as `raidLevel=5`)
- Parameter-name (such as `batteryInstallDate`)
- Operation-name (such as `redundancyCheck`)

A user-defined entry (such as user label) is called a variable. In the syntax, it is shown in italic (such as `trayID` or `volumeGroupName`).

Synopsis of the script commands

Because you can use the script commands to define and manage the different aspects of a storage array (such as host topology, drive configuration, controller configuration, volume definitions, and volume group definitions), the actual number of commands is extensive. The commands, however, fall into general categories that are reused when you apply the commands to configure or maintain a storage array. The following table lists the general form of the script commands and a definition of each command.

Syntax	Description
`accept object {statement-data}`	Performs the pending operation.
`activate object {statement-data}`	Sets up the environment so that an operation can take place or performs the operation if the environment is already set up correctly.
`autoConfigure storageArray {statement-data}`	Automatically creates a configuration that is based on the parameters that are specified in the command.
`check object {statement-data}`	Starts an operation to report on errors in the object, which is a synchronous operation.
`clear object {statement-data}`	Discards the contents of some attributes of an object. This operation is destructive and cannot be reversed.
`create object {statement-data}`	Creates an object of the specified type.
`deactivate object {statement-data}`	Removes the environment for an operation.
`delete object`	Deletes a previously created object.
`diagnose object {statement-data}`	Runs a test and shows the results.
`disable object {statement-data}`	Prevents a feature from operating.

Syntax	Description
`download object {statement-data}`	Transfers data to the storage array or to the hardware that is associated with the storage array.
`enable object {statement-data}`	Sets a feature to operate.
`load object {statement-data}`	Transfers data to the storage array or to the hardware that is associated with the storage array. This command is functionally similar to the `download` command.
`recopy object {statement-data}`	Restarts a volume copy operation by using an existing volume copy pair. You can change the parameters before the operation is restarted.
`recover object {statement-data}`	Re-creates an object from saved configuration data and the statement parameters. (This command is similar to the `create` command.)
`remove object {statement-data}`	Removes a relationship between objects.
`repair object {statement-data}`	Repairs errors found by the `check` command.
`replace object {statement-data}`	The specified object replaces an existing object in the storage array.
`reset object {statement-data}`	Returns the hardware or an object to an initial state.

Syntax	Description
`resume object`	Starts a suspended operation. The operation starts where it left off when it was suspended.
`revive object`	Forces the object from the Failed state to the Optimal state. Use this command only as part of an error recovery procedure.
`save object {statement-data}`	Writes information about the object to a file.
`set object {statement-data}`	Changes object attributes. All changes are completed when the command returns.
`show object {statement-data}`	Shows information about the object.
`start object {statement-data}`	Starts an asynchronous operation. You can stop some operations after they have started. You can query the progress of some operations.
`stop object {statement-data}`	Stops an asynchronous operation.
`suspend object {statement-data}`	Stops an operation. You can then restart the suspended operation, and it continues from the point where it was suspended.
`validate object {statement-data}`	Validates a security key.

Recurring syntax elements

Recurring syntax elements are a general category of parameters and options that you can use in the script commands. The following table lists the conventions used in the recurring syntax elements.

Convention	Definition
`a	b`	Alternative ("a" or "b")
`italicized-words`	Needs user input to fulfill a parameter (a response to a variable)
`[ ... ]` (square brackets)	Zero or one occurrence (square brackets are also used as a delimiter for some command parameters)
`{ ... }` (curly braces)	Zero or more occurrences
`(a | b | c)`	Choose only one of the alternatives

The following table lists the recurring syntax parameters and the values that you can use with the recurring syntax parameters.

Recurring Syntax	Syntax Value
autoconfigure-vols-attr- value-list	*autoconfigure-vols-attr-value-pair* {*autoconfigure-vols-attr-value-pair*}
autoconfigure-vols-attr- value-pair	```driveType=drive-type	driveMediaType=drive-media-type	raidLevel=raid-level	volumeGroupWidth=integer-literal	volumeGroupCount=integer-literal	volumesPerGroupCount=integer-literal6| hotSpareCount=integer-literal	segmentSize=segment-size-spec	cacheReadPrefetch=(TRUE | FALSE) securityType=(none | capable | enabled)7| dataAssurance=(none | enabled)5```
boolean	```(TRUE | FALSE)```
cache-flush-modifier- setting	```immediate, 0, .25, .5, .75, 1, 1.5, 2, 5, 10, 20, 60, 120, 300, 1200, 3600, infinite```

Recurring Syntax	Syntax Value
capacity-spec	*integer-literal* [KB | MB | GB | TB | Bytes]
count-based-repository- spec	`repositoryRAIDLevel =repository-raid-level repositoryDriveCount=integer-literal [repositoryVolumeGroupUserLabel =user-label] [driveType=drive-type4 ] [trayLossProtect=(TRUE | FALSE)1] | [drawerLossProtect=(TRUE | FALSE)2] | [dataAssurance=(none | enabled)5] |`
create-raid-vol-attr- value-list	*create-raid-volume-attribute-value-pair* {*create-raid-volume-attribute-value-pair*}
create-raid-volume- attribute-value-pair	`capacity=capacity-spec | owner=(a | b) | cacheReadPrefetch=(TRUE | FALSE) | segmentSize=integer-literal | usageHint=usage-hint-spec`
create-volume-copy-attr- value-list	*create-volume-copy-attr-value-pair* {*create-volume-copy-attr-value-pair*}
create-volume-copy-attr- value-pair	`copyPriority=(highest | high | medium | low | lowest) | targetReadOnlyEnabled=(TRUE | FALSE) | copyType=(offline | online) | repositoryPercentOfBase=(20 | 40 | 60 | 120 | default) | repositoryGroupPreference=(sameAsS ource | otherThanSource | default)`

Recurring Syntax	Syntax Value
drive-media-type	(HDD | SSD | unknown | allMedia) *HDD* means hard disk drive. *SSD* means solid state disk.
drive-spec	*trayID,slotID* or trayID,drawerID,slotID A drive is defined as two or three integer literal values separated by a comma. Low-density trays require two values. High-density trays, those trays that have drawers, require three values.
drive-spec-list	*drive-spec drive-spec*
drive-type	(fibre | SATA | SAS) (i) Only SAS drives are supported for firmware versions 7.86 and later.
error-action	(stop | continue)
ethernet-port-options	enableIPv4=(TRUE | FALSE) | enableIPv6=(TRUE | FALSE) | IPv6LocalAddress=ipv6-address | IPv6RoutableAddress=ipv6-address | IPv6RouterAddress=ipv6-address | IPv4Address=ip-address | IPv4ConfigurationMethod= (static | dhcp) | IPv4GatewayIP=ip-address | IPv4SubnetMask=ip-address | duplexMode=(TRUE | FALSE) | portSpeed=(autoNegotiate | 10 | 100 | 1000)
feature-identifier	(i) All features in SANtricity 11.40 are enabled by default.
filename	*string-literal*
gid	*string-literal*
hex-literal	A literal in the range of 0x00 - 0xFF.

Recurring Syntax	Syntax Value
host-card-identifier	(1 | 2 | 3 | 4)
host-type	string-literal| integer-literal
| *instance-based- repository-spec* | ```
(repositoryRAIDLevel =repository-
raid-level repositoryDrives=
(drive-spec-list)
[repositoryVolumeGroupUserLabel
=user-label]
[trayLossProtect=(TRUE | FALSE)1])
|
[drawerLossProtect=(TRUE |
FALSE)2]) |
(repositoryVolumeGroup=user-label
[freeCapacityArea=integer-
literal3])
```<br><br>Specify the **repositoryRAIDLevel** parameter with the **repositoryDrives** parameter. Do not specify the RAID level or the drives with the volume group. Do not set a value for the **trayLossProtect** parameter when you specify a volume group. |
| *ip-address* | **(0-255).(0-255).(0-255).(0-255)** |
| *ipv6-address* | **(0-FFFF):(0-FFFF):(0-FFFF):(0-FFFF):**
(0-FFFF):(0-FFFF):(0-FFFF):(0-FFFF)

You must enter all 32 hexadecimal characters. |
| *iscsi-host-port* | ```
(1 | 2 | 3 | 4)
```<br><br>The host port number might be 2, 3, or 4 depending on the type of controller you are using. |

Recurring Syntax	Syntax Value
| *iscsi-host-port-options* | ```
IPv4Address=ip-address |
IPv6LocalAddress=ipv6-address |
IPv6RoutableAddress=ipv6-address |
IPv6RouterAddress=ipv6-address |
enableIPv4=(TRUE | FALSE) |
enableIPv6=(TRUE | FALSE) |
enableIPv4Priority=(TRUE | FALSE)
|
enableIPv6Priority=(TRUE | FALSE)
| IPv4ConfigurationMethod=(static
| dhcp) |
IPv6ConfigurationMethod= (static |
auto) | IPv4GatewayIP=ip-address |
IPv6HopLimit=integer |
IPv6NdDetectDuplicateAddress=integ
er |
IPv6NdReachableTime=time-interval
| IPv6NdRetransmitTime=time-
interval |
IPv6NdTimeOut=time-interval |
IPv4Priority=integer |
IPv6Priority=integer |
IPv4SubnetMask=ip-address |
IPv4VlanId=integer |
IPv6VlanId=integer |
maxFramePayload=integer |
tcpListeningPort=tcp-port-id |
portSpeed=(autoNegotiate | 1 | 10)
``` |
| *iscsiSession* | ```
[session-identifier]
``` |
| *nvsram-offset* | *hex-literal* |
| *nvsramBitSetting* | *nvsram-mask, nvsram-value* = *0xhexadecimal, 0xhexadecimal*│*integer-literal*

The *0xhexadecimal* value is typically a value from 0x00 to 0xFF. |

Recurring Syntax	Syntax Value
nvsramByteSetting	*nvsram-value* = *0xhexadecimal* | *integer-literal* The `0xhexadecimal` value is typically a value from 0x00 to 0xFF.
portID	```(0-127)```
raid-level	```(0 | 1 | 3 | 5 | 6)```
recover-raid-volume-attr- value-list	*recover-raid-volume-attr-value-pair* {*recover-raid-volume-attr-value-pair*}

recover-raid-volume-attr- value-pair	```owner=(a|b)```
	 ```|cacheReadPrefetch=(TRUE | FALSE)```
	 ```| dataAssurance=(none | enabled)```

repository-raid-level	```(1 | 3 | 5 | 6)```
repository-spec	`instance-based-repository-spec` | `count-based-repository-spec`
segment-size-spec	*integer-literal* - all capacities are in base-2.
serial-number	```string-literal```

Recurring Syntax	Syntax Value

slotID	For high-capacity drive trays, specify the tray ID value, the drawer ID value, and the slot ID value for the drive. For low-capacity drive trays, specify the tray ID value and the slot ID value for the drive. Tray ID values are 0 to 99. Drawer ID values are 1 to 5.
	
	All slot ID maximums are 24. Slot ID values either begin at 0 or 1, depending on the tray model.
	
	Enclose the tray ID value, the drawer ID value, and the slot ID value in square brackets ([ ]).
	
	<pre>(drive=\(trayID,[drawerID,]slotID\)) |
	drives=\(trayID1,[drawerID1,]slotID1 ...
	trayIDn,[drawerIDn,]slotIDn\))</pre>

test-devices	<pre>controller=(a|b)
	esms=(esm-spec-list)drives=(drive-spec-list)</pre>

test-devices-list	*test-devices* {*test-devices*}

time-zone-spec	<pre>(GMT+HH:MM | GMT-HH:MM)
	[dayLightSaving=HH:MM]</pre>

trayID-list	*trayID* {*trayID*}
usage-hint-spec	<pre>usageHint=(multiMedia | database | fileSystem)</pre> The usage hint, or expected I/O characteristics, of the volume are used by the controller to indicate an appropriate default volume segment size and dynamic cache read prefetch. For file system and database, a 128 KB segment size is used. For multimedia, a 256 KB segment size is used. All three usage hints enable dynamic cache read prefetch.

Recurring Syntax	Syntax Value
user-label	*string-literal* Valid characters are alphanumeric, the dash, and the underscore.
user-label-list	*user-label*{*user-label*}
volumeGroup-number	*integer-literal*
wwID	*string-literal*

1For tray loss protection to work, your configuration must adhere to the following guidelines:

Level	Criteria for Tray Loss Protection	Minimum number of trays required
Disk Pool	The disk pool contains no more than two drives in a single tray.	6
RAID 6	The volume group contains no more than two drives in a single tray.	3
RAID 3 or RAID 5	Each drive in the volume group is located in a separate tray.	3
RAID 1	Each drive in a RAID 1 pair must be located in a separate tray.	2
RAID 0	Cannot achieve Tray Loss Protection.	Not applicable

2For drawer loss protection to work, your configuration must adhere to the following guidelines:

Level	Criteria for drawer loss protection	Minimum number of drawers required
Disk Pool	The pool includes drives from all five drawers and there are an equal number of drives in each drawer. A 60-drive tray can achieve Drawer Loss Protection when the disk pool contains 15, 20, 25, 30, 35, 40, 45, 50, 55, or 60 drives.	5

Level	Criteria for drawer loss protection	Minimum number of drawers required
RAID 6	The volume group contains no more than two drives in a single drawer.	3
RAID 3 or RAID 5	Each drive in the volume group is located in a separate drawer.	3
RAID 1	Each drive in a mirrored pair must be located in a separate drawer.	2
RAID 0	Cannot achieve Drawer Loss Protection.	Not applicable

If you have a storage array configuration in which a volume group spans several trays, you must make sure that the setting for drawer loss protection works with the setting for tray loss protection. You can have drawer loss protection without tray loss protection. You cannot have tray loss protection without drawer loss protection. If the **trayLossProtect** parameter and the **drawerLossProtect** parameter are not set to the same value, the storage array returns an error message and a storage array configuration will not be created.

3 To determine if a free capacity area exists, run the `show volumeGroup` command.

4 The default drive (drive type) is `SAS`.

The **driveType** parameter is not required if only one type of drive is in the storage array. If you use the **driveType** parameter, you also must use the **hotSpareCount** parameter and the **volumeGroupWidth** parameter.

5 The **dataAssurance** parameter relates to the Data Assurance (DA) feature.

The Data Assurance (DA) feature increases data integrity across the entire storage system. DA enables the storage array to check for errors that might occur when data is moved between the hosts and the drives. When this feature is enabled, the storage array appends error-checking codes (also known as cyclic redundancy checks or CRCs) to each block of data in the volume. After a data block is moved, the storage array uses these CRC codes to determine if any errors occurred during transmission. Potentially corrupted data is neither written to disk nor returned to the host.

If you want to use the DA feature, start with a pool or volume group that includes only drives that support DA. Then, create DA-capable volumes. Finally, map these DA-capable volumes to the host using an I/O interface that is capable of DA. I/O interfaces that are capable of DA include Fibre Channel, SAS, and iSER over InfiniBand (iSCSI Extensions for RDMA/IB). DA is not supported by iSCSI over Ethernet, or by the SRP over InfiniBand.

> (i) When all the required hardware and the I/O interface is DA-capable, you can set the **dataAssurance** parameter to `enabled` and then use DA with certain operations. For example, you can create a volume group that includes DA-capable drives, and then create a volume within that volume group that is DA-enabled. Other operations that use a DA-enabled volume have options to support the DA feature.

6 The **volumesPerGroupCount** parameter is the number of equal-capacity volumes per volume group.

7 The **securityType** parameter enables you to specify the security setting for a volume group that you are creating. All of the volumes are also set to the security setting that you choose. Available options for setting the security setting include:

- `none` — The volume group is not secure.

- `capable` — The volume group is security capable, but security has not been enabled.

- `enabled` — The volume group is security enabled.

> ⓘ A storage array security key must already be created for the storage array if you want to set **securityType=enabled**. (To create a storage array security key, use the `create storageArray securityKey` command.)

Naming conventions

- Names can have a maximum of 30 characters.
- You can use any combination of alphanumeric characters, hyphens, and underscores for the names of the following components:
  - Storage arrays
  - Host groups
  - Hosts
  - Volume groups
  - Dynamic disk pools
  - Volumes
  - HBA host ports
- You must use unique names. If you do not use unique names, the controller firmware returns an error.
- If the name contains more than one word, hyphens, or underscores, enclose the name in double quotation marks ("` `"). In some usages, you must also surround the name with square brackets ([ ]). The description of each parameter indicates whether you need to enclose a parameter in double quotation marks, square brackets, or both.
- The name character string cannot contain a new line.
- On Windows operating systems, you must enclose the name between two backslashes (\\) in addition to other delimiters. For example, the following name is used in a command that runs under a Windows operating system:

```
[\"Engineering\"]
```

- For a UNIX operating system and, when used in a script file, the name appears as in the following example:

```
["Engineering"]
```

- When you enter a World Wide Identifier (WWID) of an HBA host port, some usages require that you surround the WWID with double quotation marks. In other uses, you must surround the WWID with angle brackets (<>). The description of the WWID parameter indicates whether you need to enclose the WWID in double quotation marks or angle brackets.

Entering numerical names

When the storage management software automatically configures a storage array, the storage management software assigns names that consist of numerical characters. Names that consist only of numerical characters are valid names. Numerical character names, however, must be treated differently than names that start with alphabetic characters.

- Names that are only numbers, such as 1 or 2
- Names that start with a number, such as 1Disk or 32Volume
- [\"1\"]
- [\"1Disk\"]

> ⓘ If you have any doubt as to the validity of a name, use both double quotation marks and square brackets. Using both makes sure that the name works and does not cause any processing issues.

Format rules

Format rules for CLI commands

Double quotation marks (" ") that are used as part of a name or label require special consideration when you run the CLI commands and the script commands on a Windows, Linux, or Solaris operating system.

When double quotation marks (" ") are part of a name or value, you must insert a backslash (\) before each double quotation mark character.

For example:

```
-c "set storageArray userLabel=\"Engineering\";"
```

In this example, "Engineering" is the storage array name.

A second example is:

```
-n \"My\"_Array
```

In this example, "My"_Array is the name of the storage array.

You cannot use double quotation marks (" ") as part of a character string (also called string literal) within a script command. For example, you cannot enter the following string to set the storage array name to "Finance" Array:

```
-c "set storageArray userLabel=\"\"Finance\"Array\";"
```

In a Windows operating system, if you do not use double quotation marks (" ") around a name, you must insert a caret (^) before each special script character. Special characters are ^, | , <, and >.

Insert a caret before each special script character when used with the terminals `-n`, `-o`, `-f`, and `-p`. For example, to specify storage array CLI>CLIENT, enter this string:

```
-n CLI^>CLIENT
```

Insert one caret (^) before each special script character when used within a string literal in a script command. For example, to change the name of a storage array to FINANCE_|_PAYROLL, enter the following string:

```
-c "set storageArray userLabel=\"FINANCE_^|_PAYROLL\";"
```

> (i) When issuing SMcli commands at the command prompt, an additional backslash (\) character is required as an escape character in the file path name of an input or output file when using the Windows OS. The additional backslash (\) must be included when using SMcli in secure, https mode, by providing the -u option to specify a role-based access management user. **Example:**
>
> ```
> C:\\dir\\subdir\\filename
> ```

Format rules for script commands

Syntax unique to a specific script command is explained in the Notes section at the end of each script command description.

Case sensitivity — The script commands are not case sensitive. You can type the script commands in lowercase, uppercase, or mixed case. (In the following command descriptions, mixed case is used as an aid to reading the command names and understanding the purpose of the command.)

Spaces — You must enter spaces in the script commands as they are shown in the command descriptions.

Square brackets — Square brackets are used in two ways:

- As part of the command syntax.
- To indicate that the parameters are optional. The description of each parameter tells you if you need to enclose a parameter value in square brackets.

Parentheses — Parentheses shown in the command syntax enclose specific choices for a parameter. That is, if you want to use the parameter, you must enter one of the values enclosed in parentheses. Generally, you do not include parentheses in a script command; however, in some instances, when you enter lists, you must enclose the list in parentheses. Such a list might be a list of tray ID values and slot ID values. The description of each parameter tells you if you need to enclose a parameter value in parentheses.

Vertical bars — Vertical bars in a script command indicate "or" and separate the valid values for the parameter.

For example, the syntax for the `raidLevel` parameter in the command description appears as follows:

```
raidLevel=(0 | 1 | 3 | 5 | 6)
```

To use the `raidLevel` parameter to set RAID level 5, enter this value:

```
raidLevel=5
```

Drive locations — The CLI commands that identify drive locations support both high-capacity drive trays and low-capacity drive trays. A high-capacity drive tray has drawers that hold the drives. The drawers slide out of the drive tray to provide access to the drives. A low-capacity drive tray does not have drawers. For a high-capacity drive tray, you must specify the identifier (ID) of the drive tray, the ID of the drawer, and the ID of the slot in which a drive resides. For a low-capacity drive tray, you need only specify the ID of the drive tray and the ID of the slot in which a drive resides. For a low-capacity drive tray, an alternative method for identifying a location for a drive is to specify the ID of the drive tray, set the ID of the drawer to 0, and specify the ID of the slot in which a drive resides. Separate the ID values with a comma. If you enter more than one set of ID values, separate each set of values with a space.

There are two conventions for specifying drives in the CLI. The convention you should use is specified in each command. One convention uses an equal sign and parentheses:

```
drive=\(trayID,[drawerID,]slotID\)
```

The second convention uses no equal sign, but a pair of braces around the specified drives:

```
drive \[trayID,[drawerID,]slotID\]
```

Here are some examples using parentheses:

```
(1,1 1,2 1,3 1,4 2,1 2,2 2,3 2,4)
```

or, for a high-capacity drive tray, this example:

```
(1,1,1 1,2,2 1,3,3 1,4,4 2,1,1 2,2,2 2,3,3 2,4,4)
```

Italicized terms — Italicized terms in the command indicate a value or information that you need to provide. For example, when you encounter the italicized term:

```
*numberOfDrives*
```

Replace the italicized term with a value for the number of drives that you want to include with the script command.

Semicolon — Script commands must end with a semicolon (;). You can enter more than one script command on the command line or in a script file. For example, a semicolon is used to separate each script command in the following script file.

```
create volume drives=(0,2 0,3 1,4 1,5 2,6 2,7) raidLevel=5 userLabel="v1"
capacity=2gb owner=a;
create volume volumeGroup=2 userLabel="v2" capacity=1gb owner=b;
create volume volumeGroup=2 userLabel="v3" capacity=1gb owner=a;
```

```
create volume drives=(0,4 0,5 1,6 1,7 2,8 2,9) raidLevel=5 userLabel="v4"
capacity=2gb owner=b;
create volume volumeGroup=3 userLabel="v5" capacity=1gb owner=a;
create volume volumeGroup=3 userLabel="v6" capacity=1gb owner=b;
```

Format rules for CLI commands in Windows PowerShell

The Windows PowerShell is an interactive and scripting shell that provides access to command-line tools. The Windows PowerShell improves upon the Windows Command Prompt with a more robust set of commands and scripting capability. You can run all of the CLI and script commands in the Windows PowerShell; however, the Windows PowerShell has some unique formatting requirements. The requirements are these:

- Begin all SMcli commands with a period and a forward slash (./)
- SMcli wrapper must be identified as an executable command with the .exe extension (SMcli.exe)
- Enclose the script command in single quotation marks (' ')
- Double quotation marks that are part of a name, file path, or value must have a backslash before each double quotation mark character (\")

The following is an example of a CLI command to create a storage array name in the Windows PowerShell. Note the use of the single quotation marks as delimiters for the script command and the backslash double quotation marks around the storage array name, identified as `userLabel` in the command syntax.

```
PS C:\...\StorageManager\client> ./SMcli.exe 123.45.67.88 123.45.67.89
-c 'set storageArray userLabel=\"Engineering\";'
```

The following is an example of a CLI command to enable a premium feature in the Windows PowerShell. Note the use of the backslash double quotation marks before the file path to the premium feature key.

```
PS C:\...\StorageManager\client> ./SMcli.exe 123.45.67.88 123.45.67.89
-c 'enable storageArray feature file=\"C:\licenseKey.bin\";'
```

In the previous examples both upper case letters and lower case letters are used. This use is to help make clear how the commands are formatted. The Windows PowerShell is, however, not case sensitive and does not require the use specific cases.

Detailed error reporting

Data collected from an error encountered by the CLI is written to a file. Detailed error reporting under the CLI works as follows:

- If the CLI must abnormally end running CLI commands and script commands, error data is collected and saved before the CLI finishes.
- The CLI saves the error data by writing the data to a standard file name.
- The CLI automatically saves the data to a file. Special command line options are not required to save the error data.
- You are not required to perform any action to save the error data to a file.
- The CLI does not have any provision to avoid over-writing an existing version of the file that contains error data.

For error processing, errors appear as two types:

- Terminal errors or syntax errors that you might enter.
- Exceptions that occur as a result of an operational error.

When the CLI encounters either type of error, the CLI writes information that describes the error directly to the command line and sets a return code. Depending on the return code, the CLI also might write additional information about which terminal caused the error. The CLI also writes information about what it was expecting in the command syntax to help you identify any syntax errors that you might have entered.

When an exception occurs while a command is running, the CLI captures the error. At the end of processing the command (after the command processing information has been written to the command line), the CLI automatically saves the error information to a file.

The name of the file to which error information is saved is `excprpt.txt`. The CLI tries to place the `excprpt.txt` file in the directory that is specified by the system property `devmgr.datadir`. If for any reason the CLI cannot place the file in the directory specified by `devmgr.datadir`, the CLI saves the `excprpt.txt` file in the same directory from which the CLI is running. You cannot change the file name or the location. The `excprpt.txt` file is overwritten every time that an exception occurs. If you want to save the information in the `excprpt.txt` file, you must copy the information to a new file or a new directory.

Exit status

This table lists the exit statuses that might be returned and the meaning of each status.

Status Value	Error Name	Meaning
0	SUCCESSFUL	The command terminated without an error.
1	INVALID_USAGE	The command terminated with an error. Information about the error also appears.

Status Value	Error Name	Meaning
2	SCRIPT_FILE_DOES_NOT_EXIST	The script file does not exist.
3	ERROR_OPENING_OUTPUT_FILE	An error occurred while opening an output file.
4	NO_STORAGE_ARRAY_AT_ADDRESS	A storage array was not at the specified address.
5	ADDRESSES_SPECIFY_DIFFERENT_ARRAYS	Addresses specify different storage arrays.
6	NO_SANAME_FOR_HOST_AGENT_CONNECT	A storage array name does not exist for the host agent that is connected.
7	SANAME_NOT_AT_ADDRESS	The storage array name was not at the specified address.
8	SANAME_NOT_UNIQUE	The storage array name was not unique.
9	SANAME_NOT_IN_CONFIG_FILES	The storage array name was not in the configuration file.
10	NO_MANAGEMENT_CLASS_FOR_SA	A management class does not exist for the storage array.
11	NO_SA_IN_CONFIG_FILE_FOUND	A storage array was not found in the configuration file.
12	INTERNAL_ERROR	An internal error occurred. This exit status indicates that you do not have privileges for running a CLI command from the command line. You must have administrator privileges to run all of the CLI commands from a command line.
13	INVALID_SCRIPT_SYNTAX	Invalid script syntax was found.
14	UNABLE_TO_COMMUNICATE	The controller was unable to communicate with the storage array.
15	DUPLICATE_ARGUMENT	A duplicate argument was entered.

Status Value	Error Name	Meaning
16	EXECUTION_ERROR	An execution error occurred.
17	NO_HOST_AT_ADDRESS	A host was not at the specified address.
18	WWNAME_NOT_IN_CONFIG_FILES	The WWID was not in the configuration file.
19	WWNAME_NOT_AT_ADDRESS	The WWID was not at the address.
20	UNKNOWN_IP	An unknown IP address was specified.
21	PM_CONFIG_FILE_CORRUPTED	The Event Monitor configuration file was corrupted.
22	UNABLE_TO_COMMUNICATE_TO_PM	The storage array was unable to communicate with the Event Monitor.
23	UNEXIST_ALERT	The controller was unable to write alert settings.
24	WRONG_ORGANIZER_NODE	The wrong organizer node was specified.
25	CMD_NOT_AVAILABLE	The command was not available.
26	DEVICE_NOT_IN_CONFIG_FILE	The device was not in the configuration file.
27	ERROR_UPDATING_CONFIG_FILE	An error occurred while updating the configuration file.
28	UNKNOWN_HOST_ERROR	An unknown host error occurred.
29	SENDER_CONTACT_FILE_NOT_FOUND	The sender contact information file was not found.
30	READ_SENDER_CONTACT_FILE_FAILED	The sender contact information file could not be read.
31	USERDATA_FILE_EXISTS	The `userdata.txt` file exists.

Status Value	Error Name	Meaning
32	BAD_EMAIL_INFORMATION_TO_ INCLUDE	An invalid $-I$ value in the email alert notification was specified.
33	BAD_EMAIL_FREQUENCY	An invalid $-f$ value in the email alert notification was specified.
34	OPTION_REMOVED	The $-r$ option is not supported anymore.
35	UNKNOWN_ALERT_PRIORITY	Invalid alert severity specified.
36	PASSWORD_REQUIRED	The operation needs either the Administrator or Monitor password to be set.
37	INVALID_MONITOR_PASSWORD	The operation cannot be completed because an invalid Monitor password was entered.
38	INVALID_ADMIN_PASSWORD	The operation cannot be completed because an invalid Administrator password was entered.
39	EXCEEDED_MAX_CHARS_FOR_ PASSWORD	The password provided is exceeding the character limit.
40	INVALID_MONITOR_TOKEN	The $-R$ monitor is not supported for this array. Use a valid role and retry the operation.
41	ASUP_CONFIG_ERROR	An error occurred while writing to or reading from the AutoSupport configuration file. Please retry this operation again.
42	MAIL_SERVER_UNKNOWN	Host address or mail server address is incorrect.
43	ASUP_SMTP_REPLY_ADDRESS _REQUIRED	No healthy ASUP enabled arrays detected while attempting ASUP configuration test.
44	NO_ASUP_ARRAYS_DETECTED	Reply email request required if ASUP delivery type is SMTP.

Status Value	Error Name	Meaning
45	ASUP_INVALID_MAIL_RELAY_SERVER	Unable to validate ASUP mail relay server.
46	ASUP_INVALID_SENDER_EMAIL	Sender email address you specified is not a valid format.
47	ASUP_INVALID_PAC_SCRIPT	Proxy Auto-Configuration (PAC) script file is a not a valid URL.
48	ASUP_INVALID_PROXY_SERVER_HOST_ADDRESS	The Host address specified cannot be found or is in an incorrect format.
49	ASUP_INVALID_PROXY_SERVER_PORT_NUMBER	The port number you specified is an invalid format.
50	ASUP_INVALID_AUTHENTICATION_PARAMETER	The username or password you specified is invalid.
51	ASUP_INVALID_DAILY_TIME_PARAMETER	The daily time parameter specified is invalid.
52	ASUP_INVALID_DAY_OF_WEEK_PARAMETER	The `-dayOfWeek` parameter(s) that have been entered are invalid.
53	ASUP_INVALID_WEEKLY_TIME_PARAMETER	The weekly time parameter is not valid.
54	ASUP_INVALID_SCHEDULE_PARSING	Unable to successfully parse the schedule information provided.
55	ASUP_INVALID_SA_SPECIFIED	Invalid storage array specifier provided.
56	ASUP_INVALID_INPUT_ARCHIVE	The input archive entered is invalid. The input archive parameter must be in the form of `-inputArchive=<n>`where `-n` is an integer from 0 to 5.
57	ASUP_INVALID_OUTPUT_LOG	A valid output log was not specified.

Status Value	Error Name	Meaning
58	ASUP_TRANSMISSION_FILE_COPY_ERROR	There was an error trying to copy the AutoSupport transmission log file. The transmission log either does not exist or there was an IO error trying to copy its data.
59	ASUP_DUPLICATE_NAMED_ARRAYS	More than one storage array with the same name was found. Please retry the command using the world-wide-name parameter, `-w <WWID>`.
60	ASUP_NO_SPECIFIED_ARRAY_FOUND	The specified storage array with the -n <storage-system-name> parameter is not present or is not supported for this command.
61	ASUP_NO_SPECIFIED_WWID_FOUND	The specified storage array with the `-w <WWID>` parameter is not present or is not supported for this command.
62	ASUP_FILTERED_TRANSMISSION_LOG_ERROR	There was an unknown error trying to obtain the filtered transmission log.
63	ASUP_TRANSMISSION_ARCHIVE_DOES_NOT_EXIST	The AutoSupport input archive transmission log specified with the `-inputArchive=<n>` parameter does not exist.
64	NO_VALID_REST_CLIENT_DISCOVERED	Unable to communicate with the storage array via https.
65	INVALID_CLI_VERSION	The client CLI version is not compatible with the CLI version running on the storage array.
66	INVALID_USERNAME_OR_PASSWORD	The username or password entered is invalid.
67	UNTRUSTED_CONNECTION	Unable to establish a secure connection to the storage array.
68	INVALID_PASSWORD_FILE	The password file cannot be found or is not readable.

Add comments to a script file

The script engine looks for certain characters or a command to show comments. You can add comments to a script file in three ways:

- Add text after two forward slashes (//) as a comment until an end-of-line character is reached. If the script engine does not find an end-of-line character in the script after processing a comment, an error message appears, and the script operation is terminated. This error usually occurs when a comment is placed at the end of a script and you have forgotten to press the **Enter** key.

  ```
  // Deletes the existing configuration.
  set storageArray resetConfiguration=true;
  ```

- Add text between / and / as a comment. If the script engine does not find both a starting comment notation and an ending comment notation, an error message appears, and the script operation is terminated.

  ```
  /* Deletes the existing configuration */
  set storageArray resetConfiguration=true;
  ```

- Use the show statement to embed comments in a script file that you want to appear while the script file is running. Enclose the text that you want to appear by using double quotation marks ("` `").

  ```
  show "Deletes the existing configuration";
  set storageArray resetConfiguration=true;
  ```

Script command guidelines

This list provides guidelines for writing script commands on the command line:

- You must end all commands with a semicolon (;).
- You can enter more than one command on a line, but you must separate each command with a semicolon (;).
- You must separate each base command and its associated primary parameters and secondary parameters with a space.
- The script engine is not case sensitive. You can enter commands by using uppercase letters, lowercase letters, or mixed-case letters.
- Add comments to your scripts to make it easier for you and future users to understand the purpose of the script commands. (For information about how to add comments, see Adding comments to a script file.)

> ℹ️ While the CLI commands and the script commands are not case sensitive, user labels (such as for volumes, hosts, or host ports) are case sensitive. If you try to map to an object that is identified by a user label, you must enter the user label exactly as it is defined, or the CLI commands and the script commands will fail.

Firmware compatibility levels

The script commands and the command parameters do not run under all versions of the controller firmware. The script commands in the following sections list the minimum firmware levels under which the script commands can run. In the script commands, the firmware levels are listed under the heading "Minimum Firmware Level." This list describes how to interpret the information about the firmware levels.

- If a script command does not list a minimum controller firmware level, the script command and all of the parameters associated with that script command can run under any level of controller firmware.

- A controller firmware number without any explanatory information indicates that the controller firmware level applies to the entire script command and all of the parameters for that script command.

- A controller firmware number that is associated with a parameter indicates the minimum controller firmware level under which the parameter can run.

> (i) The minimum controller firmware level indicates support by the software that releases the command, as well as support by all storage management software that picks up usage. CLI support capabilities depend on the hardware used. When an unsupported command is entered, an error message appears.

Examples of firmware compatibility levels

The `create hostGroup` command has the following section.

Minimum firmware level

8.10

This level indicates that the entire script command runs under a minimum of controller firmware version 8.10.

The `create ssdCache` command has the following section.

Minimum firmware level

7.84

8.20.11 - Added the `securityType` parameter.

These notations indicate that the script command and all of the parameters except `securityType` run under a minimum of controller firmware version 7.84. The `securityType` parameter runs under a minimum of controller firmware version 8.20_M3.