

# Adaptec Serial ATA RAID 2810SA Command Line Interface

Software Reference Guide



**Copyright**

© 2000 - 2003 Adaptec, Inc. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written consent of Adaptec, Inc., 691 South Milpitas Blvd., Milpitas, CA 95035.

**Trademarks**

Adaptec, Adaptec Storage Manager, and the Adaptec logo are trademarks of Adaptec, Inc., which may be registered in some jurisdictions.

Windows 2000 and Windows XP are trademarks of Microsoft Corporation in the US and other countries, used under license.

All other trademarks are the property of their respective owners.

**Changes**

The material in this document is for information only and is subject to change without notice. While reasonable efforts have been made in the preparation of this document to assure its accuracy, Adaptec, Inc. assumes no liability resulting from errors or omissions in this document, or from the use of the information contained herein.

Adaptec reserves the right to make changes in the product design without reservation and without notification to its users.

**Disclaimer**

IF THIS PRODUCT DIRECTS YOU TO COPY MATERIALS, YOU MUST HAVE PERMISSION FROM THE COPYRIGHT OWNER OF THE MATERIALS TO AVOID VIOLATING THE LAW WHICH COULD RESULT IN DAMAGES OR OTHER REMEDIES.

# Contents

## **1 Getting Started**

- Audience 1-2
- Supported Platforms 1-2
- How to Use this Manual 1-2
- Organization 1-3
- Conventions 1-3
- Related Documentation 1-4

## **2 Accessing the Command Line Interface**

- Accessing the CLI from the MS-DOS Prompt 2-2
- Accessing the CLI from the Windows Start Button 2-2
- Accessing the CLI from the NetWare Server Console 2-3
- Accessing the CLI from the Linux/UNIX Prompt 2-3

## **3 Syntax and Conventions for CLI Commands**

- CLI Features 3-1
- CLI Command Syntax 3-2
  - Class 3-3
  - Action 3-3
  - Object 3-3
  - Switch 3-3
  - Value 3-4
  - Parameter 3-4
  - Blank Space 3-4
- CLI Parameter and Switch Value Types 3-4
  - boolean 3-5

- integer 3-5
- string 3-6
- scsi\_device 3-7
- free\_space 3-8
- container 3-9
- CLI Status Information 3-10
  - The Stat and Task Items 3-10
  - The Func Item 3-11
  - The Ctr and State Items 3-12

## **4 General Control Commands**

- close 4-2
  - Syntax 4-2
- exit 4-2
  - Syntax 4-2
- help, ? 4-2
  - Syntax 4-2
  - Parameters 4-2
  - Switches 4-2
- history\_size 4-3
  - Command Availability 4-3
  - Syntax 4-3
  - Parameters 4-3
- open 4-3
  - Syntax 4-3
  - Parameters 4-3
  - Switches 4-4
- reset\_window 4-5
  - Command Availability 4-5
  - Syntax 4-5
- toggle\_more 4-5
  - Command Availability 4-5
  - Syntax 4-5

## **5 container Commands**

- container add\_level 5-3

Syntax 5-3	
Parameters 5-3	
Examples 5-3	
Related Information 5-4	
container create mirror 5-5	
Syntax 5-5	
Parameters 5-5	
Switches 5-6	
Examples 5-6	
Related Information 5-8	
container create mmirror 5-9	
Notes 5-9	
Syntax 5-9	
Parameters 5-9	
Switches 5-10	
Examples 5-11	
Related Information 5-12	
container create mstripe 5-14	
Notes 5-14	
Syntax 5-14	
Parameters 5-15	
Switches 5-15	
Examples 5-16	
Related Information 5-18	
container create mvolume 5-19	
Notes 5-19	
Syntax 5-19	
Parameters 5-19	
Switches 5-20	
Examples 5-20	
Related Information 5-23	
container create raid5 5-24	
Notes 5-24	
Syntax 5-24	
Parameters 5-24	
Switches 5-25	
Examples 5-27	

**Related Information 5-29**

`container create stripe` 5-30

Notes 5-30

Syntax 5-30

Parameters 5-30

Switches 5-31

Examples 5-32

**Related Information 5-34**

`container create volume` 5-35

Notes 5-35

Syntax 5-35

Parameters 5-35

Switches 5-36

Examples 5-37

**Related Information 5-38**

`container delete` 5-40

Notes 5-40

Syntax 5-40

Parameters 5-40

Switches 5-40

Examples 5-41

**Related Information 5-42**

`container extend file_system` 5-43

Notes 5-43

Command And Switch Availability 5-44

Syntax 5-44

Parameters 5-44

Examples 5-44

**Related Information 5-45**

`container extend mvolume` 5-46

Command And Switch Availability 5-46

Syntax 5-46

Parameters 5-46

Examples 5-47

**Related Information 5-49**

`container extend volume` 5-50

Command And Switch Availability 5-50

	Syntax 5-50
	Parameters 5-50
	Examples 5-51
	Related Information 5-53
<code>container list</code>	5-54
	Syntax 5-54
	Parameters 5-54
	Switches 5-54
	Examples 5-55
	Related Information 5-64
<code>container lock</code>	5-65
	Syntax 5-65
	Parameters 5-65
	Examples 5-65
	Related Information 5-66
<code>container move</code>	5-67
	Notes 5-67
	Syntax 5-67
	Parameters 5-67
	Examples 5-68
	Related Information 5-68
<code>container promote</code>	5-69
	Notes 5-69
	Syntax 5-69
	Parameters 5-69
	Examples 5-70
	Related Information 5-71
<code>container readonly</code>	5-72
	Syntax 5-72
	Parameters 5-72
	Examples 5-72
	Related Information 5-73
<code>container readwrite</code>	5-74
	Syntax 5-74
	Parameters 5-74
	Examples 5-74
	Related Information 5-75

- container reconfigure 5-76
  - Notes 5-77
  - Syntax 5-77
  - Parameters 5-78
  - Switches 5-78
  - Examples 5-81
  - Related Information 5-82
- container release\_cache 5-83
  - Syntax 5-83
  - Parameters 5-83
  - Examples 5-83
  - Related Information 5-83
- container remove drive\_letter 5-84
  - Command Availability 5-84
  - Syntax 5-84
  - Parameters 5-84
  - Examples 5-84
  - Related Information 5-85
- container remove failover 5-86
  - Syntax 5-86
  - Parameters 5-86
  - Examples 5-87
  - Related Information 5-87
- container remove file\_system 5-88
  - Command Availability 5-88
  - Syntax 5-88
  - Parameters 5-88
  - Examples 5-88
  - Related Information 5-89
- container remove global\_failover 5-90
  - Syntax 5-90
  - Parameters 5-90
  - Examples 5-90
  - Related Information 5-91
- container restore RAID5 5-92
  - Syntax 5-92
  - Parameters 5-92



- Examples 5-92
- Related Information 5-93
- `container scrub` 5-94
  - Syntax 5-94
  - Parameters 5-94
  - Switches 5-94
  - Examples 5-95
  - Related Information 5-96
- `container set cache` 5-97
  - Notes 5-97
  - Syntax 5-97
  - Parameters 5-98
  - Switches 5-98
  - Examples 5-98
  - Related Information 5-99
- `container set failover` 5-100
  - Syntax 5-100
  - Parameters 5-100
  - Examples 5-100
  - Related Information 5-101
- `container set global_failover` 5-102
  - Syntax 5-102
  - Parameters 5-102
  - Examples 5-102
  - Related Information 5-103
- `container set io_delay` 5-104
  - Syntax 5-104
  - Parameters 5-104
  - Examples 5-105
  - Related Information 5-106
- `container set label` 5-107
  - Syntax 5-107
  - Parameters 5-107
  - Examples 5-107
  - Related Information 5-108
- `container show cache` 5-109
  - Syntax 5-109

- Parameters 5-109
- Examples 5-109
- Related Information 5-110
- `container show failover` 5-112
  - Syntax 5-112
  - Parameters 5-112
  - Examples 5-112
  - Related Information 5-112
- `container split` 5-113
  - Syntax 5-113
  - Parameters 5-114
  - Examples 5-114
  - Related Information 5-116
- `container unlock` 5-117
  - Syntax 5-117
  - Parameters 5-117
  - Examples 5-117
  - Related Information 5-118
- `container unmirror` 5-119
  - Syntax 5-119
  - Parameters 5-119
  - Examples 5-119
  - Related Information 5-120

## **6 controller Commands**

- `controller details` 6-2
  - Syntax 6-2
  - Examples 6-2
  - Related Information 6-5
- `controller firmware compare` 6-6
  - Syntax 6-6
  - Switches 6-6
  - Examples 6-6
  - Related Information: 6-6
- `controller firmware save` 6-7
  - Syntax 6-7
  - Switches 6-7

- Examples 6-7
- Related Information: 6-8
- controller firmware update 6-9
  - Syntax 6-9
  - Switches 6-9
  - Examples 6-9
  - Related Information: 6-9
- controller list 6-10
  - Syntax 6-10
  - Parameters 6-10
  - Switches 6-10
  - Examples 6-10
  - Related Information 6-11
- controller pause\_io 6-12
  - Command And Switch Availability 6-13
  - Notes 6-13
  - Syntax 6-13
  - Parameters 6-13
  - Examples 6-13
  - Related Information 6-14
- controller rescan 6-15
  - Syntax 6-15
- controller reset\_scsi\_channel 6-16
  - Syntax 6-16
  - Parameters 6-16
  - Examples 6-16
- controller resume\_io 6-17
  - Syntax 6-17
  - Examples 6-17
  - Related Information 6-17
- controller set automatic\_failover 6-18
  - Notes 6-18
  - Syntax 6-18
  - Examples 6-19
  - Related Information 6-19
- controller set array\_verify 6-20
  - Syntax 6-20
  - Examples 6-20

**Related Information 6-21**

`controller show automatic_failover` 6-22

**Syntax 6-22**

**Examples 6-22**

**Related Information 6-22**

`controller show channels` 6-23

**Syntax 6-23**

**Examples 6-23**

**Related Information 6-24**

`controller show array_verify` 6-25

**Syntax 6-25**

**Examples 6-25**

**Related Information 6-25**

## **7 disk Commands**

`disk blink` 7-2

**Syntax 7-2**

**Parameters 7-2**

**Switches 7-2**

**Examples 7-2**

`disk initialize` 7-3

**Syntax 7-3**

**Parameters 7-3**

**Switches 7-3**

**Examples 7-4**

`disk list` 7-5

**Syntax 7-5**

**Parameters 7-5**

**Switches 7-5**

**Examples 7-5**

**Related Information 7-8**

`disk remove dead_partitions` 7-9

**Syntax 7-9**

**Parameters 7-9**

**Examples 7-9**

**Related Information 7-9**

`disk set default` 7-10

	Syntax 7-10
	Parameters 7-10
	Examples 7-10
	Related Information 7-11
disk set smart	7-12
	Syntax 7-12
	Parameters 7-12
	Switches 7-13
	Examples 7-14
	Related Information 7-14
disk show partition	7-15
	Syntax 7-15
	Examples 7-15
disk show smart	7-18
	Syntax 7-18
	Parameters 7-18
	Switches 7-18
	Examples 7-19
	Related Information 7-21
disk show space	7-22
	Syntax 7-22
	Parameters 7-22
	Switches 7-22
	Examples 7-22
disk verify	7-25
	Syntax 7-25
	Parameters 7-25
	Switches 7-25
	Examples 7-26
disk zero	7-27
	Command And Switch Availability 7-27
	Syntax 7-27
	Parameters 7-27
	Switches 7-27
	Examples 7-28

## **8 logfile Commands**

logfile end 8-2

Syntax 8-2

Examples 8-2

Related Information 8-2

logfile start 8-3

Syntax 8-3

Parameter 8-3

Switches 8-3

Examples 8-3

Related Information 8-4

## **9 task Commands**

task list 9-2

Syntax 9-2

Parameters 9-2

Switches 9-2

Examples 9-2

The Specific2 Column 9-6

Related Information 9-7

task resume 9-8

Syntax 9-8

Parameters 9-8

Switches 9-8

Examples 9-8

Related Information 9-9

task stop 9-10

Syntax 9-10

Parameters 9-10

Switches 9-10

Examples 9-10

Related Information 9-11

task suspend 9-12

Syntax 9-12

Parameters 9-12

Switches 9-12

Examples 9-12

Related Information 9-13

## **A Automated Scripts**

Creating an Automated Script A-1

# 1

## Getting Started

### In this Chapter

<i>Audience</i>	1-2
<i>Supported Platforms</i>	1-2
<i>How to Use this Manual</i>	1-2
<i>Organization</i>	1-3
<i>Conventions</i>	1-4
<i>Related Documentation</i>	1-4

The Adaptec SCSI RAID controller integrates advanced RAID technology with state-of-the-art controller hardware to deliver leadership performance, data availability, and storage management features.

To configure and manage components running on the Adaptec SCSI RAID controller, Adaptec has developed an easy-to-use graphical user interface (GUI) called Adaptec Storage Manager as well as a command line interface (CLI).

The CLI provides a rich alternative to Adaptec Storage Manager. Through the CLI, you perform most of the storage management tasks that you can perform with the Adaptec Storage Manager GUI and, in addition, some tasks not available in the GUI. The CLI also enables you to use the CLI commands in DOS command scripts and Linux/UNIX shell scripts.



This reference guide describes each of the commands available in the CLI.

## **Audience**

This reference guide is written for system administrators and experienced users who are familiar with device configuration and who have a general understanding of Microsoft Windows NT<sup>®</sup> 4.0 / Windows 2000 (basic disk) / Windows XP (basic disk). If you are running under NetWare, the guide assumes you are familiar with NetWare-related concepts and commands.

## **Supported Platforms**

The Adaptec SCSI RAID controller operates on a variety of operating system platforms. See the installation guide that was supplied with your controller for a list of the supported operating system platforms.

Unless otherwise indicated, the commands and switches discussed in this guide are supported on the operating systems listed in the installation guide. For the latest supported operating system versions of the Adaptec SCSI RAID controller, visit Adaptec's product web site at [www.adaptec.com](http://www.adaptec.com).

## **How to Use this Manual**

The Adaptec Advanced RAID Command Line Interface utility is intended for use with both Adaptec SCSI and SATA RAID controllers. This utility was originally created for SCSI controllers and consequently many commands and their output use SCSI terminology. The most significant differences for a SATA controller are: The number of buses and devices is not always reported correctly The device ID is interpreted differently for SATA devices Some commands are intended specifically for SCSI devices and do not function on SATA devices. These problems do not affect the basic functionality of the CLI utility or your ability to create and manage arrays or the controller using the CLI.

This manual documents software originally written to support SCSI RAID controllers. With the introduction of Serial ATA RAID

technology, the software has been adapted by mapping of ports 0 through 3 to the following SCSI IDs:

- Bus 0, ID 0, Lun 0
- Bus 0, ID 1, Lun 0
- Bus 0, ID 2, Lun 0
- Bus 0, ID 3, Lun 0

In general, SCSI IDs will be populated from Bus 0, ID 0 upwards (always LUN 0).

## Organization

Chapter 1, *Getting Started*, provides an overview of this reference guide.

Chapter 2, *Accessing the Command Line Interface*, provides information on the ways you can access the CLI.

Chapter 3, *Syntax and Conventions for CLI Commands*, describes the features, conventions, command syntax, parameters, and value types that define the command line interface.

Chapter 4, *General Control Commands*, details general purpose commands such as `open` and `close`.

Chapter 5, *container Commands*, describes all the commands related to arrays.

Chapter 6, *controller Commands*, describes all the commands related to controllers.

Chapter 7, *disk Commands*, describes the commands related to disks.

Chapter 8, *logfile Commands*, describes the commands used to record CLI command activities.

Chapter 9, *task Commands*, describes the commands used to work with tasks.

Appendix A, *Automated Scripts*, provides examples of how to use the CLI commands in DOS command scripts.

## Conventions

The following typographical conventions are used in this reference guide.

<b>bold</b>	Used to highlight information depicted in examples (for example, the <b>Type</b> column indicates that a RAID-5 set resides on array 0).
Courier	Used to highlight commands in text (for example, the <code>array create mirror</code> command).

## Related Documentation

Use this reference guide when you need specific information about a CLI command (for example, syntax and parameter descriptions).



**Note:** When consulting the *Command Line Interface Reference Guide*, you can consider the terms “array” and “container” as synonymous if you keep in mind that containers have features that no other arrays have; such as, using the full capacity of a disk when creating a container and sharing a disk’s full capacity across multiple containers.

In addition, the term “array” has a general meaning beyond “RAID array;” for example, a rack of disks is often referred to as a disk array.

Other related documents include the following:

■ *Adaptec Serial ATA RAID Installation & User’s Guide*

The installation guide that ships with your RAID controller explains how to install it in your computer system. The installation guide also provides instructions on how to install the RAID software.

Describes the features available in Adaptec Storage Manager and how to use them.

The guide assumes that you are familiar with:

- Basic functions of your operating system
- Basic network administration terminology and tasks
- RAID technology

# 2

## Accessing the Command Line Interface

### In this Chapter

<i>Accessing the CLI from the MS-DOS Prompt</i>	2-2
<i>Accessing the CLI from the Windows Start Button</i>	2-2
<i>Accessing the CLI from the NetWare Server Console</i>	2-3
<i>Accessing the CLI from the Linux/UNIX Prompt</i>	2-3



---

**Note:** When the CLI executes an asynchronous command, it displays status information in the title bar of a DOS command prompt window. The status information includes a percent number that indicates percentage completion. On NetWare, the status information appears in a different console screen. On UNIX, the status information appears at the bottom of the CLI window.

---

## Accessing the CLI from the MS-DOS Prompt

To access the CLI from the MS-DOS prompt, move to the directory containing the `afaccli` executable and then type `afaccli`.



**Note:** To access the CLI from any location, the path in the Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only) Registry or System Variables must include the directory where the CLI executable is installed. Or, you must be in the directory where the CLI executable resides.

See your Windows NT or MS-DOS documentation for more information.

## Accessing the CLI from the Windows Start Button

You can execute the command line interface (CLI):

- From the Windows XP, Windows 2000 or Windows NT Start button



**Note:** The following procedure assumes that you accepted the default location for the software during installation.

To access the CLI from the Windows XP (basic disk only), Windows 2000(basic disk only) or Windows NT Start button:

- 1 Click the **Start** button and move the mouse cursor to **Programs**. Click on **Programs**.
- 2 Move the mouse cursor to **SMBE**. The Adaptec program group will display.
- 3 Move the mouse cursor to the **CLI** icon. Click on **CLI**.

- From the NetWare server console

To access the CLI from the NetWare server console, type `aaccli` at the prompt. When the system displays the `CLI>` prompt, which indicates that you can now use CLI commands.

- From the UNIX/Linux prompt

To access the CLI from the UNIX/Linux prompt, display a window and type `aaccli` in any directory. When the system displays the `CLI>` prompt, you can use CLI commands. For the commands to work in any directory, the path in the startup file (`.login` or `.cshrc`) must include the directory where the software is installed. See your UNIX/Linux documentation for information on setting up directory paths in the `.login` and `.cshrc` files.

## Accessing the CLI from the NetWare Server Console

To access the CLI from the NetWare server console, type `aaccli` at the prompt. The system displays the `CLI>` prompt, which indicates that you can now use CLI commands.

## Accessing the CLI from the Linux/UNIX Prompt

To access the CLI from the UNIX/Linux prompt, display a window and type `aaccli` in any directory. The system displays the `CLI>` prompt, which indicates you can now use CLI commands. For the command to work in any directory, the path in the startup file (`.login` or `.cshrc`) must include the directory where the software is installed. See your UNIX documentation for information on setting up directory paths in the `.login` and `.cshrc` files.

# 3

## Syntax and Conventions for CLI Commands

In this Chapter

<i>CLI Features</i>	3-1
<i>CLI Command Syntax</i>	3-2
<i>CLI Parameter and Switch Value Types</i>	3-4
<i>CLI Status Information</i>	3-10

### CLI Features

To use the CLI efficiently, you need to understand the following features:

- **Case sensitivity**—CLI syntax elements are not case-sensitive except for strings.
- **Command abbreviations**—You can abbreviate commands, subcommands, and switches. You must provide enough characters so the commands are not ambiguous.
- **Subcommand Mode**—Subcommand mode simplifies command entry when you are performing repetitive commands on the same item. Subcommand mode works only for commands that have one or more subcommands.

To enter subcommand mode, type the first part of any command that has subcommands and press **Enter**. The command you entered replaces the CLI> prompt and becomes the first part of any subsequent command you enter.

For example, to use subcommand mode for the `array` command:

- 1 Type `array` at the CLI> prompt, and press **Enter**.

The CLI> prompt changes to `_array >`.

- 2 Enter valid subcommands, switches, and parameters for any `array` command and press **Enter**. For example, to change the number of array 1 to 2, simply type the following at the `_array >` prompt and press **Enter**:

```
move 1 2
```

To exit subcommand mode, press **Enter**. (You may have to press **Enter** more than once to exit subcommand mode.)

#### ■ CLI-unique commands

For NetWare systems, the following commands are unique to the CLI in that you use these commands when accessing the CLI from the NetWare server console:

- `open`
- `close`
- `exit`
- `toggle_more`

- **Comments**—If the first non-blank character on a line in the CLI is an exclamation point (!), the rest of the line will be ignored. You can type comments after the exclamation point. This is particularly useful in scripts where you want to comment out sections of code.

## CLI Command Syntax

You can enter only valid CLI software commands at the command line prompt.



All commands use the following syntax:

```
[class] action [object] [/switch1[=value1]
[/switch2[=value2]...]] [parameter1 [parameter2...]]
```

The CLI syntax uses the following conventions:

[text]	The text element within brackets is optional.
{type}	The type specified within braces is the value type (for example, boolean, integer, string, and so on).
parameter{type}	The parameter defined by the value type.
[text...]	Two or more text elements.

## Class

CLI commands are grouped according to *class*. Classes currently supported include array, controller, disk, logfile, and task.

## Action

An *action* specifies an operation to be performed by the CLI, such as open, create, list, exit, or show. An action is a required element, except in subcommand mode. In most cases, actions are preceded by a class and followed by an object, switches, and parameters.

## Object

An *object* describes what to use in an action. Examples include mirror, volume, mstripe, and drive\_letter.

## Switch

A *switch* is an element applied to a command that allows a variation of the command. Switches are always optional.

If you specify a switch, it may require a value. Switches that are Boolean do not require a value when the switch is used because the switch defaults to `TRUE` or `FALSE`. Optionally, you can explicitly set a Boolean switch to `TRUE` or `FALSE`.

## Value

A *value* is a type that applies to a switch. For each switch, you can have only *one* type of value. For example, a switch that is defined as a string value can never take an integer value. A space may be used within a value only within a quoted string. Examples of valid value types include Boolean, integer, string, `scsi_device`, `free_space`, and array.

You delimit a value that applies to a switch by using an equal sign, as in the following example:

```
AAC0>disk list /all =TRUE
Executing: disk list /all=TRUE
```

Note that a space to the left or right of the equal sign is permitted.

## Parameter

Although some commands have optional parameters, *parameters* are usually mandatory. You must place optional parameters after mandatory parameters at the end of the command. For example, the `array remove failover` command has one required `scsi_device` parameter and additional optional `scsi_device` parameters: `array remove failover {array} {scsi_device} [{scsi_device}...]`.

As with switch values, parameters can be of only *one* type. For example, the `array` parameter can take only an integer value.

All parameters are separated by blank space. A space can be used in a parameter only within a quoted string.

## Blank Space

*Blank space* is one or more spaces or tabs.

## CLI Parameter and Switch Value Types

A parameter or switch value can be only one of several types. Simple value types include string, Boolean, and integer. Other value types include objects that can be manipulated, such as arrays (or arrays) and devices. This section explains the CLI parameter and switch value types.

## boolean

A *boolean* has a value of either `TRUE` or `FALSE` and can be specified with any of the following key words:

- `TRUE` can be specified with `TRUE`, `1`, `ON`, or `YES`. For switches, you can also specify `TRUE` by using the switch without any switch value. For example, `/readonly` and `/readonly=TRUE` are identical.
- `FALSE` can be specified with `FALSE`, `0`, `OFF`, or `NO`. When a switch is not specified, it takes the default value (usually, `FALSE`) specified in the switch description section of a command. For example, `open aac0` is the same as `open /readonly=FALSE aac0`.

## integer

An *integer* is a positive or negative number that has a value between  $-2^{63}$  and  $(2^{63} - 1)$ . Although the valid range is usually much smaller, all integers have 64 bits of precision and do not contain decimal points.

You can specify an integer as a mathematical equation that uses an asterisk (\*) to multiply, a plus sign (+) to add, a minus sign (-) to subtract, a slash (/) to divide, and parentheses (()) to specify order of operations. If you do not use parentheses, all operations are completed left to right. No spaces are allowed in the expression.

Numbers in an equation that:

- Begin and end with no suffix are decimal
- Begin with `0x` or end with `h` are hexadecimal
- End with `o` indicate octal
- End with `z` indicate binary

You can attach special multipliers to the end of any number to allow for easy translation to reasonable disk sizes. Table 3-1 lists the letters and their multiplicative values:

Table 3-1. Letters and Multiplicative Values

Letter	Action
K (kilobytes)	multiplies by 1024
M (megabytes)	multiplies by 1024*1024
G (gigabytes)	multiplies by 1024*1024*1024
T (terabytes)	multiplies by 1024*1024*1024*1024

You cannot use decimal points. To specify 1.5 GB, for example, you must use (3G/2).



**Note:** All suffixes are case-insensitive! That is, you can use upper or lower-case characters. For example you can specify 10 M or 10 m.

Table 3-2 lists examples of valid integers and their corresponding values:

Table 3-2. Integers and Values (in Decimal)

Integer	Value (in decimal)
219	219
3*4	12
(5+3*24)	192
(5+(3*24))	77
0x123	291
(12+52h+1010z)	104
100M	104,857,600
2G	2,147,483,648

## string

You can specify a text *string* with or without quotation marks. If spaces are needed, however, the string must be specified with quotation marks; otherwise, the CLI interprets spaces as delimiters.

A string can use either double quotes or single quotes but must start and end with the same type of quotes. Also, within a quoted string,

a quotation mark of the same type used to surround the string can be specified by repeating that quotation mark twice in the string.

Table 3-3 lists examples of valid strings and their corresponding values:

Table 3-3. Strings and Corresponding Values

String	Value
VOL_set	VOL_set
"VOL label"	VOL label
'Use " mark'	Use " mark
"Use ' mark"	Use ' mark
"Use Both "" and ' mark"	Use Both " and ' mark

### `scsi_device`

Specifies a device. Device descriptions have three parts: SCSI channel number, device ID, and device logical unit number (LUN).

The following syntax defines the `scsi_device` parameter:

```
( {integer} , {integer} , {integer} )
```

Each integer corresponds to one component of a device descriptor or specifier: the first `integer` is the SCSI channel number, the second `integer` is the device ID, and the third `integer` is the device LUN.

The actual values that the CLI supports are as follows:

- Channel number = (for example, 0, 1, 2, 3, and so on). See the installation guide for your controller to determine the actual number of channels it supports.
- Device ID = (0 through 15 inclusive)
- LUN = 0 through 7 inclusive

Table 3-4 displays the syntax for device switch value type abbreviations:

Table 3-4. SCSI Device Switch Abbreviations

Abbreviation	Syntax
SCSI channel number, SCSI device ID	<b>{integer},{integer}</b>
SCSI device ID	<b>{integer}</b>
SCSI device ID	<b>{integer}</b>

Table 3-5 displays valid SCSI device specifiers. The second two examples contain a default LUN number, and the last two examples contain default channel and LUN numbers. You can specify these defaults using the `disk set default` command (see [page 7-10](#)).

Table 3-5. SCSI Device Specifiers

Example	SCSI Channel Number, SCSI Device ID, SCSI Device LUN
(1,1,4)	Channel 1, Device 1, LUN 4
(3,2,0)	Channel 3, Device 2, LUN 0
(3,2)	Channel 3, Device 2, LUN default <sup>1</sup>
(0,1)	Channel 0, Device 1, LUN default <sup>1</sup>
(1)	Channel default <sup>1</sup> , Device 1, LUN default <sup>1</sup>
1	Channel default <sup>1</sup> , Device 1, LUN default <sup>1</sup>

<sup>1</sup> Set by the `disk set default` command.

## free\_space

Freespace (also known as *available space*) is specified by a SCSI device and, optionally, a size. If you do not specify a size, parentheses are optional.

Therefore, the following syntaxes are allowed for the `free_space` parameter:

```
( {scsi_device} , {freespace_size} )
( {scsi_device} )
{scsi_device}
```

If you do not specify the `freespace_size` parameter, it defaults to the size of the first freespace area available on the specified SCSI device. Note that offsets are not specified for freespace. The offset

used is the first offset that starts a freespace area large enough to fit size bytes.

Table 3-6 lists valid `free_space` specifiers.

Table 3-6. Freespace Specifiers

Freespace Specifier	SCSI Channel Number, SCSI Device ID, SCSI Device LUN; Use <i>n</i> Amount of Freespace
<code>((1,2),1G)</code>	SCSI Device: Channel 1, Device ID 2, LUN default; use 1 GB of freespace
<code>5</code>	SCSI Device: Channel default, Device ID 5, LUN default; use all freespace if empty or all of the first freespace available is some is used
<code>(8,2G)</code>	SCSI Device: Channel default, Device ID 8, LUN default; use 2 GB of freespace



**Note:** When specifying freespace during an array create volume operation, you must explicitly specify the SCSI device's channel, device ID, and logical unit number. You cannot use any default values for the SCSI device. In addition, you must also specify the size of the freespace(s).

## container

A controller currently supports 24 visible arrays. A visible array is an array that is visible to the operating system and users. Visible arrays are identified with array IDs 0 through 23.

Array IDs 24 through 63 are reserved for hidden arrays. A hidden array is an array that is not visible to the operating system and can only be used by other arrays. (However, the `array list` command displays hidden arrays.)

The Parameters section uses the following syntax to specify an array:

```
{array}
```

This syntax specifies the ID number (0 to 63 inclusive) of the array.

A controller assigns a unique ID to each of its arrays when you create it.

## **CLI Status Information**

When invoked from the MS-DOS prompt or the Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only) Start button, the CLI displays status information in the title bar of a DOS command prompt window as it executes an asynchronous command. On NetWare, the status information appears in a different window. On UNIX systems, the CLI displays this status information at the bottom of the CLI window.

The following example shows the status information for the `array create mirror` command:

```
Stat:OK!Task:101,Func:MCR Ctr:0,State:DNE 100.0%
```

The following sections describe each item that the CLI displays in the title bar.

### **The Stat and Task Items**

The Stat item displays the status of the currently running task. Typically, the item displays the value `OK!` to indicate the task is executing correctly. This is the value that appears in the example.

The Task item displays the ID number associated with a specific task. The controller software assigns each task a unique ID number. The task ID that appears in the example is 101.



## The Func Item

The Func item displays the type of task running on the controller. Table 3-7 describes the values that the Func item can display.

Table 3-7. Function Values

Value	Meaning
FSV	File system verify task.
FSX	File system extend task. The <code>array extend mvolume</code> and <code>array extend volume</code> commands cause the <code>FSX</code> value to display.
FTF	array format task with a FAT file system specified. The <code>array format</code> command with the <code>/file_system</code> switch set to <code>FAT</code> causes the <code>FTF</code> value to display.
MCR	Mirror set create or multilevel mirror set create task. The <code>array create mirror</code> and <code>array create mmirror</code> commands cause the <code>MCR</code> value to display.
MMR	Merge a broken mirror task. The <code>array merge</code> command causes the <code>MMR</code> value to display.
MSC	array scrub task. The <code>array scrub</code> command causes the <code>MSC</code> value to display.
NTF	array format task with an NTFS file system specified. The <code>array format</code> command with the <code>/file_system</code> switch set to <code>NTFS</code> causes the <code>NTF</code> value to display.
R5R	RAID-5 rebuild task.
R5S	RAID-5 set create task with the scrub method specified. The <code>array create raid5</code> command with the <code>/scrub</code> switch specified causes the <code>R5S</code> value to display.
RCF	array reconfigure task. The <code>array reconfigure</code> command causes the <code>RCF</code> value to display.
SCV	Verify all blocks on a SCSI disk device task. The <code>disk verify</code> command causes the <code>SCV</code> value to display.
SCZ	Clear an entire SCSI disk task. The <code>disk zero</code> command causes the <code>SCZ</code> value to display.
SVR	Verify all blocks and repair bad blocks on a SCSI disk device task. The <code>disk verify</code> command with the <code>/repair</code> switch causes the <code>SVR</code> value to display.

## The Ctr and State Items

The Ctr item displays the ID number of the array associated with the task. In the example, the array's ID is 0 (zero).

The State item displays the state of the task along with a running percentage value that indicates the progress of the currently running task. The item shows the percentage in tenths of a percent increments. The currently running task is complete when the State item displays 100%.

Table 3-8 describes the state values that the State item can display.

Table 3-8. State Values

State	Meaning
BAD	The task failed and is no longer running.
DNE	The task successfully completed or the task is no longer running (that is, the task is done).
RUN	The task is running.
SUS	The task was suspended. Typically, you suspend a task with the <code>task suspend</code> command.
UNK	The controller reported an unknown status for the task.

# 4

## General Control Commands

### In this Chapter

<i>close</i>	4-2
<i>exit</i>	4-2
<i>help, ?</i>	4-2
<i>history_size</i>	4-3
<i>open</i>	4-3
<i>reset_window</i>	4-5
<i>toggle_more</i>	4-5

The CLI general control commands are discussed in alphabetical order and use the following syntax:

```
command [subcommand] [/switch{=value}] [parameter]
```

## **close**

To close the currently opened controller when all access is completed, use the `close` command.

### **Syntax**

```
close
```

## **exit**

To close the currently opened controller and exit the CLI, use the `exit` command.

### **Syntax**

```
exit
```

## **help, ?**

To invoke general or topical Help commands, use the `help` command or the `?` (question mark).

### **Syntax**

```
? [{command}]
```

```
{command} ?
```

```
help [/full] [{command}]
```

```
help [/full] {command subset}
```

### **Parameters**

If the command has more subcommands, `help` lists the subcommands and their functions. If a complete command is used, such as `array list`, the CLI help displays all possible switches.

### **Switches**

```
/full
```

Displays all relevant commands along with the command format and all command switches.

## history\_size

To set the size of the command history buffer, use the `history_size` command.

### Command Availability

This command is supported only on UNIX.

### Syntax

```
history_size {buffer_size}
```

### Parameters

{buffer\_size}

Specifies the size of the command history buffer. The default size is 200.

## open

To open a controller, use the `open` command. The `open` command prepares a particular controller for access by the CLI. If you specify this command when another controller is open during a particular command session, the CLI closes the currently opened controller and then opens the specified controller.

### Syntax

```
open [/readonly{=boolean}]  
[/domain{=string}] {string}
```

### Parameters

{string}

Specifies the computer name and the controller you want to open. For the string, use the standard format `\\nodename\AACn`, where *n* is the controller number.

For local controllers, you can omit the `\\nodename` in the string specification.

If the computer name has a dash (-) in the name (for example, proj-athena), enclose the entire string within quotes. For example:

```
"\\proj-athena\aac0"
```

## Switches

/readonly{=boolean}

Specifies whether to open the controller for read-only access. A value of `TRUE` indicates the CLI opens the controller for read-only access.

If you do not specify this switch, it defaults to /  
`readonly=FALSE` (which means the CLI opens the controller for read/write access).

Note that if you open the controller with read-only access, you can use only the commands that do not change the controller configuration.

/domain{=string}

Specifies the domain (the local domain or a trusted domain) in which the specified computer that contains the controller resides. If you do not specify this switch, the CLI assumes the local domain.

This switch is supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)



---

**Note:** You can open controllers for read-write access in any GUI or CLI session only once per controller. Thereafter, you (and other users) can open and access the previously opened controller only in read-only mode.

---

## **reset\_window**

To reset the window, use the `reset_window` command.

### **Command Availability**

This command is supported only on UNIX.

### **Syntax**

```
reset_window
```

## **toggle\_more**

To turn on or off the <Press any key to continue> functionality, use the `toggle_more` command.

After you type `toggle_more` on the NetWare console, the command displays an appropriate message indicating whether the <Press any key to continue> functionality is on or off.

### **Command Availability**

This command is supported only on NetWare.

### **Syntax**

```
toggle_more
```

# 5

## container Commands

### In this Chapter

<i>container add_level</i>	5-3
<i>container create mirror</i>	5-5
<i>container create mmirror</i>	5-9
<i>container create mstripe</i>	5-14
<i>container create mvolume</i>	5-19
<i>container create raid5</i>	5-24
<i>container create stripe</i>	5-30
<i>container create volume</i>	5-35
<i>container delete</i>	5-40
<i>container extend file_system</i>	5-43
<i>container extend mvolume</i>	5-46
<i>container extend volume</i>	5-50
<i>container list</i>	5-54
<i>container lock</i>	5-65
<i>container move</i>	5-67
<i>container promote</i>	5-69
<i>container readonly</i>	5-72
<i>container readwrite</i>	5-74
<i>container reconfigure</i>	5-76
<i>container release_cache</i>	5-83



## *container Commands*

<i>container remove drive_letter</i>	5-84
<i>container remove failover</i>	5-86
<i>container remove file_system</i>	5-88
<i>container remove global_failover</i>	5-90
<i>container restore RAID5</i>	5-92
<i>container scrub</i>	5-94
<i>container set cache</i>	5-97
<i>container set failover</i>	5-100
<i>container set global_failover</i>	5-102
<i>container set io_delay</i>	5-104
<i>container set label</i>	5-107
<i>container show cache</i>	5-109
<i>container show failover</i>	5-112
<i>container split</i>	5-113
<i>container unlock</i>	5-117
<i>container unmirror</i>	5-119

The array commands are discussed in alphabetical order and use the following syntax:

```
container action [object] [/switch{=value}]  
[parameter]
```



**Note:** The examples in this chapter assume that you have first opened the controller before executing the container command. The initial step of opening the controller is not shown.

---

## container add\_level

To create a multilevel volume set from an existing array by adding a volume set on top of the array, use the `container add_level` command. The existing array cannot be a multilevel array.

The `container add_level` command enables dynamic expansion of an array and can be executed even while an array is in use. After executing the command, you can use the `container extend mvolume` command to add more space.

### Syntax

```
container add_level {container}
```

### Parameters

{container}

Specifies the ID number (0 to 63) of the array to convert to a multilevel volume set.

The specified array must not be a multilevel array.

### Examples

Before creating a multilevel volume set, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a stripe set) on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe		Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L	Offset:Size
0	Stripe	45MB	32KB	None	0:02:0	64.0KB: 15.0MB
					0:03:0	64.0KB: 15.0MB
					0:04:0	64.0KB: 15.0MB

The following example creates a multilevel volume set on top of array 0, which is a stripe set previously created with the `container create stripe` command:

```
AAC0>container add_level 0
```

```
Executing: container add_level 0
```

Use the `container list` command after using the `container add_level` command to display information about the multilevel volume set, as in the following example:

```
AAC0>container list

Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
0	Volume	45MB		None	
63	Stripe	45MB	32KB		
				0:02:0	64.0KB: 15.0MB
				0:03:0	64.0KB: 15.0MB
				0:04:0	64.0KB: 15.0MB

The following list describes the change to the display as a result of creating a multilevel volume set with the `container add_level` command:

- The **Num Label** column displays two ID numbers. The first ID (0) is the newly created volume set. The second ID (63) is the original array (the stripe set).

If you specified a label when creating the multilevel volume set, it appears in this column. Because no label was specified when the multilevel volume set was created, no label appears in the column.

- The **Type** column displays two array types: Volume and Stripe. The Volume array type indicates that the newly created array is a volume set. This is the array created after the `container add_level` command completes.

The Stripe array type indicates the array whose ID was specified to the `container add_level` command. This is the original array previously created with the `container create stripe` command.

## Related Information

container commands:

- `container extend mvolume` ([page 5-46](#))
- `container list` ([page 5-54](#))
- `container promote` ([page 5-69](#))

## container create mirror

To create a mirror set from a single-partition volume set and freespace, use the `container create mirror` command. Any data on the original volume set remains intact during mirror creation and the user sees no interruption in service.

If you created a mirror set on a NetWare server, you can run the `list devices` command on the NetWare console to verify its creation and then create the necessary NetWare disk partitions and volumes.

For information on how to create partitions and volumes, see the appropriate NetWare documentation.

### Syntax

```
container create mirror [/wait{=boolean}]  
[/io_delay{=integer}] {container} {scsi_device}
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array on which to create a mirror set. You create a mirror set from a single-partition volume set.

`{scsi_device}`

Specifies the ID for the SCSI device whose freespace you want to use for mirroring the volume set specified in the `container` parameter. A SCSI ID consists of a SCSI bus number (e.g., 0, 1, 2, 3, etc.), SCSI device ID (0 through 15 inclusive), and SCSI device logical unit number (0 through 7 inclusive). See the installation guide for your controller to determine the number of buses it actually supports.

See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

## Switches

`/wait{=boolean}`

Specifies whether to create the mirror set synchronously. If you set this switch to `TRUE`, the command prompt does not return until the mirror-set creation completes. If you set this switch to `FALSE`, the mirror-set creation starts asynchronously and the command prompt returns immediately.

`/io_delay{=integer}`

Specifies the number of milliseconds the controller waits between the I/Os required to create the mirror set. If you do not specify this switch, the I/O delay is always zero (0). The I/O delay value is not preserved between reboots of the operating system.

## Examples

Before creating a mirror set, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is one existing array (array 0, a volume set) on this controller prior to the time the mirror set is created:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume	10MB		NTFS	0:02:0 64.0KB: 10.0MB

The following example shows how to create a mirror set synchronously from volume set 0 using freespace on SCSI device (0,3,0):

```
AAC0>container create mirror /wait /io_delay=10 0
(0,3,0)
```

```
Executing: container create mirror /wait=TRUE
/io_delay=10 0
(CHANNEL=0, ID=3,LUN=0)
```

As the command executes, note the title bar of the DOS window displays the status of the command. For example:

```
Stat:OK!Task:101,Func:MCR Ctr:0,State:RUN 97.2%
```

See [page 3-10](#) for detailed information on the status information that the command displays in the title bar as it executes the task.

Use the `container list` command after the `container create mirror` command completes execution to display information about the mirror set, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth Stripe	Scsi	Partition
Dr Label Type Size Ctr Size Usage C:ID:L Offset:Size				
0	Mirror	10MB	NTFS	0:02:0 64.0KB: 10.0MB 0:03:0 64.0KB: 10.0MB

The following list describes the changes to the display as a result of using the `container create mirror` command:

- The **Type** column displays a new array type, which in the example is **Mirror** instead of the previously displayed **Volume**.
- The **State** column displays **Normal** (instead of a blank) as the mirror state.

(Note that this column appears only if you specify the `/full` switch with the `container list` command.)

- The **Scsi C: ID: L** column displays the SCSI device IDs for the two halves of the mirror set.
- The **Partition Offset:Size** column displays the partition offset and size for the two halves of the mirror set.

If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column changes to a “!” (exclamation point). See the `disk remove dead_partitions` ([page 7-9](#)) command for more information on dead partitions.

After creating a mirror set, you can manipulate it by using the

- `container split` command to split a mirror set
- `container unmirror` command to unmirror a mirror set

## **Related Information**

container commands:

- container create mmirror ([page 5-9](#))
- container list ([page 5-54](#))
- container split ([page 5-113](#))
- container unmirror ([page 5-119](#))

disk commands:

- disk remove dead\_partitions ([page 7-9](#))

## container create mmirror

To create a multilevel array of mirror sets from a multilevel array of single-partition volume sets and freespace, use the `container create mmirror` command. Typically, you use this command after promoting a volume set built from multiple partitions with the `container promote` command.

The `container create mmirror` command is fully dynamic. If you use this command, users will not see any change or experience any interruption in service. However, performance may be reduced.

### Notes

When creating a multilevel array of mirror sets on a NetWare server, you specify the container IDs (as described in the Parameters section) for the arrays from which you want to create the multilevel mirror set. If NetWare is using any of these arrays, an appropriate message displays. This message indicates that one or more of the arrays you specified is in use by NetWare. The message directs you to remove any NetWare volumes and partitions from these arrays. You can then create a multilevel array of mirror sets on a NetWare server.

After you create a multilevel array of mirror sets you can run the `list devices` command on the NetWare console to verify its creation and then create the necessary NetWare disk partitions and volumes.

For information on how to create partitions and volumes, see the appropriate NetWare documentation.

### Syntax

```
container create mmirror [/io_delay{=integer}] [/wait{=boolean}] {container} {scsi_device}
[ {scsi_device} ... ]
```

### Parameters

{container}

Specifies the ID number (0 to 63) of the array whose underlying volume sets the command converts to mirror sets. You create a multilevel array of mirror sets from a multilevel array of single-partition volume sets and freespace.



```
{scsi_device}
```

Specifies the ID for the SCSI device whose freespace you want to use to create the multilevel array of mirror sets. The size of this freespace should be greater than or equal to the size of the first underlying volume set. A SCSI ID consists of a SCSI bus number (e.g., 0, 1, 2, 3, etc.), SCSI device ID (0 through 15 inclusive), and SCSI device logical unit number (0 through 7 inclusive). See the installation guide for your controller to determine the number of buses it actually supports.

See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

```
{scsi_device}...
```

Specifies the IDs for additional devices whose freespaces you want to use to create the multilevel array of mirror sets. There must be sufficient contiguous space available on each specified device.

The number of devices you specify must be equal to the number of single-partition volume sets that make up the multilevel array of volume sets. The size of the freespace used is equal to its corresponding underlying volume set.

## Switches

```
/io_delay{=integer}
```

Specifies the number of milliseconds the controller waits between the I/Os required to create the multilevel array of mirror sets. If you do not specify this switch, the I/O delay is always zero (0). The I/O delay value is not preserved between reboots of the operating system.

```
/wait{=integer}
```

Specifies whether to create the multilevel array of mirror sets synchronously or asynchronously. If you set this switch to `TRUE`, the command creates the multilevel array of mirror sets synchronously and the command prompt does not return until the mirror-set creation task completes.

If you do not specify this switch, the mirror-set creation starts asynchronously and the command prompt returns immediately.

## Examples

Before creating a multilevel array of mirror sets, use the `container list` command to obtain information about any existing arrays. As the following example shows, array 0 is a volume set that consists of two underlying volume sets created with the `container promote` command:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume	30MB		NTFS	
63	Volume	15MB		0:02:0	64.0KB: 15.0MB
62	Volume	15MB		0:03:0	64.0KB: 15.0MB

The following example shows how to create a multilevel array of mirror sets with the following characteristics:

- The array specified (0) is a multilevel array of volume sets that was previously created with the `container promote` command.
- The freespace from devices (0,4,0), and (0,5,0) are used to mirror the underlying volume sets.
- The default I/O delay is taken.
- The multilevel array of mirror sets is created asynchronously.

```
AAC0>container create mmirror 0 (0,4,0) (0,5,0)
```

```
Executing: container create mmirror 0
```

```
(CHANNEL=0,ID=4,LUN=0) (CHANNEL=0,ID=5,LUN=0)
```

As the command executes, note the title bar of the DOS window displays the status of the command. For example:

```
Stat:OK!Task:101,Func:MCR Ctr:63,State:RUN 97.2%
```

See [page 3-10](#) for detailed information on the status information that the command displays in the title bar as it executes the task.

Use the `container list` command after using the `container create mmirror` command to display information about the multilevel array of mirror sets:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
-----	-------	-----	--------	------	-----------

Dr	Label	Type	Size	Ctr	Size	Usage	C:ID:L	Offset:Size
-----								
F: 0		Volume	30M			NTFS		
	63	Mirror	15MB				0:02:0	64.0KB: 15.0MB
							0:04:0	64.0KB: 15.0MB
	62	Mirror	15MB				0:03:0	64.0KB: 15.0MB
							0:05:0	64.0KB: 15.0MB

The following list describes the change to the display as the result of creating a multilevel array of mirror sets with the `container create mmirror` command:

- The **Dr** column is blank to indicate that the newly created mirror sets (63 and 62) do not have a drive letter assigned to them.
- The **Type** column displays Mirror for arrays 63 and 62 to indicate that the underlying arrays are mirror sets.
- The **State** column displays the state of the arrays, which in this example are all in the Normal state.

Note that this column appears only if you specify the `/full` switch with the `container list` command.

- The **Scsi C:ID:L** column displays the SCSI device ID for the disk(s) on which the underlying mirror set(s) reside. In the example, mirror set 63 resides on disk 0:02:0 and 0:04:0. Mirror set 62 resides on disk 0:03:0 and 0:05:0.
- The **Partition Offset:Size** column displays the partition offset(s) and size(s) for the underlying mirror set(s). In the example mirror set 63 and 62 have partition offsets of 64.0 KB and sizes of 15.0 MB.

If a partition is dead, the ":" (colon) in the **Partition Offset:Size** column changes to a "!" (exclamation point). See the `disk remove dead_partitions` ([page 7-9](#)) command for more information on dead partitions.

## Related Information

container commands:

- `container create mirror` ([page 5-5](#))
- `container list` ([page 5-54](#))
- `container promote` ([page 5-69](#))

disk commands:

- `disk remove dead_partitions` ([page 7-9](#))

## container create mstripe

To create a multilevel stripe set from equally sized arrays, use the `container create mstripe` command. The top level of a multilevel stripe array can only be a stripe set. You can create the following types of multilevel stripe set:

- Stripe set of mirror sets
- Stripe set of volume sets
- Stripe set of stripe sets
- Stripe set of RAID-5 sets (a RAID-50 set)

### Notes

When creating a multilevel array of stripe sets on a NetWare server, you specify the container IDs (as described in the Parameters section) for the arrays from which you want to create the multilevel stripe set. If NetWare is using any of these arrays, an appropriate message displays. This message indicates that one or more of the arrays you specified is in use by NetWare. The message directs you to remove any NetWare volumes and partitions from these arrays. You can then create a multilevel array of stripe sets on a NetWare server.

After you create a multilevel array of stripe sets you can run the `list devices` command on the NetWare console to verify its creation and then create the necessary NetWare disk partitions and volumes.

For information on how to create partitions and volumes, see the appropriate NetWare documentation.

### Syntax

```
container create mstripe [/stripe_size{=integer}]  
[/label{=string}]{container} [{container}...]
```

## Parameters

`{container}`

Specifies the ID number (0 to 63) of the array from which you want to create the multilevel stripe set.

`{container}...`

Specifies the ID number or numbers (0 to 63) of one or more arrays from which you want to create the multilevel stripe set. You can use up to 16 arrays to create a multilevel stripe set. All specified arrays must be the same size as the first `container` parameter and separated by blank spaces.

## Switches

`/stripe_size{=integer}`

Specifies the stripe size for the multilevel stripe set. Valid values are 16 KB, 32 KB, and 64 KB.

If you do not specify the switch, it defaults to 64 KB.

`/label{=string}`

Specifies a label to be assigned to the newly created multilevel stripe set. You can specify a maximum of sixteen characters for the label.

If you do not specify the switch, it defaults to no label. If you do not specify a label, you can do so later by using the `container set label` command.

## Examples

Before creating a multilevel stripe set, use the `container list` command to obtain information about any existing arrays.

As the following example shows, there are two existing arrays (mirror sets) on this controller at the time the multilevel stripe set is created. These mirror sets were previously created with the `container create mirror` command.

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type Size Ctr Size Usage C:ID:L Offset:Size					
0	Mirror 15MB			None	0:02:0 64.0KB: 15.0MB
					0:02:0 15.0MB: 15.0MB
1	Mirror 15MB			None	0:03:0 64.0KB: 15.0MB
					0:03:0 15.0MB: 15.0MB

The following example shows how to create a multilevel stripe set from two equally sized arrays, using the default stripe size of 64 KB. In this example, the two equally sized arrays are mirror sets:

```
AAC0>container create mstripe 0 1
```

```
Executing: container create mstripe 0 1
```

```
container 0 created
```

On UNIX systems, the message displayed after you execute the `container create mstripe` command includes the root special file associated with the newly created multilevel stripe set.

Use the `container list` command after using the `container create mstripe` command to display information about the multilevel stripe set, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type Size Ctr Size Usage C:ID:L Offset:Size					
0	Stripe 30MB	0	64KB	None	
63	Mirror 15MB				0:02:0 64.0KB: 15.0MB
					0:02:0 15.0MB: 15.0MB
62	Mirror 15MB				0:03:0 64.0KB: 15.0MB
					0:03:0 15.0MB: 15.0MB

The following list describes the change to the display as the result of creating a multilevel stripe set with the `container create mstripe` command:

- The **Num Label** column displays the ID number (in the example, 0) of the newly created multilevel stripe set and (indented to the right) the ID numbers of the arrays that make up the multilevel stripe set. In the example, arrays 63 and 62 are the arrays (mirror sets) that make up the multilevel stripe set.

If you specify a label when creating the multilevel stripe set, it appears in this column. Because no label was specified when the multilevel stripe set was created, no label appears in the column.

On UNIX systems, the root special file associated with the multilevel stripe set also appears in this column.

- The **Type** column displays the type Stripe for the newly created multilevel stripe set. The **Type** column also displays the array type for the underlying arrays. In the example, Mirror appears in the **Type** column to indicate that the arrays that make up the multilevel volume set are mirror sets.
- The **Total Size** column displays the total size for the multilevel stripe set. This size is the total of all the underlying arrays. In the example, the total size for array 0 (the multilevel stripe set) is the sum of the two underlying arrays (30 MB).

The **Total Size** column also displays the sizes of all the underlying arrays. In the example, the size of array 63 (15.0 MB) and array 62 (15.0 MB) were specified when the mirror sets were created with the `container create mirror` command.

- The **Stripe Size** column displays the stripe size specified for the multilevel stripe set. In this example, the display shows 32 KB, the default stripe size.
- The **Usage** column displays None to indicate that the newly created multilevel stripe set does not have a file system on it. To create an NTFS or FAT file system on an array, use the `container format` command.



- The **State** column displays Normal for arrays 0, 63, and 62 to indicate that the state of the array is normal.

Note that this column appears only if you specify the `/full` switch with the `container list` command.

- The **Scsi C:ID:L** column displays the SCSI device ID for the disk on which you created the multilevel stripe set. It also displays the SCSI device ID(s) for the disk(s) on which the array(s) that make up the multilevel stripe set reside. In the example, the display shows that array 63's partitions are on SCSI disk 0:02:0. array 62's partitions are on SCSI disk 0:03:0.
- The **Partition Offset: Size** column displays the partition offset and the size of the arrays that make up the newly created multilevel stripe set. In the example, the display shows that array 63 has partition offsets of 64.0 KB and 15.0 MB and sizes of 15.0 MB. array 62 has partition offsets of 64.0 KB and 15.0 MB and sizes of 15.0 MB. You specify the partition offset and size when you create the array.

## Related Information

container commands:

- `container create mirror` ([page 5-5](#))
- `container list` ([page 5-54](#))
- `container set label` ([page 5-107](#))

## container create mvolume

To create a multilevel volume set, use the `container create mvolume` command. You can create the following types of multilevel volume sets:

- Volume set of stripe sets
- Volume set of mirror sets
- Volume set of RAID-5 sets
- Volume set of volume sets

### Notes

When creating a multilevel array of volume sets on a NetWare server, you specify the container IDs (as described in the Parameters section) for the arrays from which you want to create the multilevel volume set. If NetWare is using any of these arrays, an appropriate message displays. This message indicates that one or more of the arrays you specified is in use by NetWare. The message directs you to remove any NetWare volumes and partitions from these arrays. You can then create a multilevel array of volume sets on a NetWare server.

After you create a multilevel volume set you can run the `list devices` command on the NetWare console to verify its creation and then create the necessary NetWare disk partitions and volumes.

For information on how to create partitions and volumes, see the appropriate NetWare documentation.

### Syntax

```
container create mvolume [/label{=string}]  
{container} [{container}...]
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array from which you want to create the multilevel volume set.

```
{container}...
```

Specifies the ID (0 to 63) of one or more additional arrays from which you want to create the multilevel volume set. You can use up to 16 arrays. Typically, the additional arrays you specify are the same type as the first array. For example, if the first array is a stripe set then any subsequent arrays are also stripe sets.

## Switches

```
/label{=string}
```

Specifies a label to be assigned to the newly created multilevel volume set. You can specify a maximum of sixteen characters for the label.

If you do not specify the switch, it defaults to no label. If you do not specify a label, you can do so later by using the `container set label` command.

## Examples

Before creating a multilevel volume set, use the `container list` command to obtain information about any existing arrays.

As the following example shows, there are two existing arrays (stripe sets) on this controller at the time the multilevel volume set is created:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition			
Dr	Label	Type	Size	Ctr	Size	Usage	C:ID:L	Offset:Size
0		Stripe	45MB		64KB	None	0:02:0	64.0KB: 15.0MB
							0:03:0	64.0KB: 15.0MB
							0:04:0	64.0KB: 15.0MB
1		Stripe	15MB		64KB	None	0:05:0	64.0KB: 15.0MB

The following example shows how to create a multilevel volume set from two single arrays:

```
AAC0> container create mvolume 0 1

Executing: container create mvolume 0 1
container 0 created
```

On UNIX systems, the message displayed after you execute the `container create mvolume` command includes the root special file associated with the newly created multilevel volume set.

Use the `container list` command after using the `container create mvolume` command to display information about the multilevel array of volume sets after you create it, as in the following example:

```
AAC0>container list

Executing: container list
```

Num	Total	Oth	Stripe	Usage	Scsi	Partition
Dr Label Type Size	Ctr	Size			C:ID:L	Offset:Size
0	Volume	60.0MB		None		
63	Stripe	45MB	64KB		0:02:0	64.0KB: 15.0MB
					0:03:0	64.0KB: 15.0MB
					0:04:0	64.0KB: 15.0MB
62	Stripe	15MB	64KB		0:05:0	64.0KB: 15.0MB

The following list describes the columns that contain information as a result of creating a multilevel volume set with the `container create mvolume` command:

- The **Num Label** column displays the ID number (in the example, 0) of the newly created multilevel volume set and (indented to the right) the ID numbers of the arrays that make up the multilevel volume set. In the example, arrays 63 and 62 are the stripe sets that make up the multilevel volume set.

If you specify a label when creating the multilevel volume set, it appears in this column. Because no label was specified when the multilevel volume set was created, no label appears in the column.

On UNIX systems, the root special file associated with the multilevel volume set also appears in this column.

- The **Type** column displays the type name **Volume** for the newly created multilevel volume set. The **Type** column also displays the array type for the underlying arrays. In the example, **Stripe** appears in the **Type** column to indicate that the arrays that make up the multilevel volume set are stripe sets.
- The **Total Size** column displays the total size for the multilevel volume set. This size is the total of all the underlying arrays. In the example, the total size for array 0 (the multilevel volume set) is the sum of the two underlying arrays (60 MB).

The **Total Size** column also displays the sizes of all the underlying arrays. In the example, the size of array 63 (45 MB) and array 62 (15 MB) were specified when the stripe sets were created with the `container create stripe` command.
- The **Stripe Size** column displays the stripe size for each of the arrays that make up the multilevel volume set. You specify the stripe size when you create stripe sets and RAID-5 sets. In the example, the stripe size is 32 KB.
- The **Usage** column displays **None** to indicate that the newly created multilevel volume set does not have a file system on it. To create an NTFS or FAT file system on an array, use the `container format` command.
- The **Scsi C:ID:L** column displays the SCSI device ID for the disk on which you created the multilevel volume set. It also displays the SCSI device ID(s) for the disk(s) on which the array(s) that make up the multilevel volume set reside. In the example, the display shows that array 63's partitions reside on three SCSI disks: 0:02:0, 0:03:0, and 0:04:0. array 62's partition resides on SCSI disk 0:05:0.
- The **Partition Offset: Size** column displays the partition offset and the size of the arrays that make up the newly created multilevel volume set. In the example, the display shows that array 63 has a partition offset of 64 KB and a size of 15 MB on each of the three SCSI disks. array 62 has a partition offset of 64 KB and a size of 15 MB. You specify the partition offset and size when you create the underlying arrays.

## **Related Information**

container commands:

- container create `mmirror` ([page 5-9](#))
- container create `raid5` ([page 5-24](#))
- container create `stripe` ([page 5-30](#))
- container set `label` ([page 5-107](#))

## container create raid5

To create a RAID-5 set, use the `container create raid5` command. The RAID-5 set must have a minimum of three disks.

When creating the RAID-5 set, you must initialize the array parity by using the scrub method (the `/scrub` switch).

### Notes

If you created a RAID-5 set you can run the `list devices` command on the NetWare console to verify its creation and then create the necessary NetWare disk partitions and volumes.

For information on how to create partitions and volumes, see the appropriate NetWare documentation.

### Syntax

```
container create raid5 [/cache{=boolean}]
[/clear{=boolean}] [/stripe_size{=integer}]
[/label{=string}] [/scrub{=boolean}]
[/wait{=boolean}] {free_space} [{scsi_device}...]
```

### Parameters

`{free_space}`

Specifies the SCSI device and its associated freespace used to create the RAID-5 set. See the *free\_space* section on page 3-8 for more information on how to specify the freespace.

`{scsi_device}...`

Specifies one or more devices. A SCSI ID consists of a SCSI bus number (e.g., 0, 1, 2, 3, etc.), SCSI device ID (0 through 15 inclusive), and SCSI device logical unit number (0 through 7 inclusive). See the installation guide for your controller to determine the number of buses it actually supports.

See the *scsi\_device* section on page 3-7 for more information on how to specify the SCSI device.

The command uses the freespace(s) from the SCSI device(s) you specify to create the RAID-5 set. The size that the command uses from these device(s) is identical to the one you specify in the `free_space` parameter. The minimum number of partitions in a RAID-5 set is 3, and the maximum number of partitions is 16.

## Switches

`/cache{=boolean}`

When `FALSE` (the default), the command causes the controller to disable the array's array cache. `FALSE` means:

- Disables the read-ahead cache.
- Disables the write-back cache.

When `TRUE`, write cache is enabled, providing maximum performance.

### Notes

- When the write cache is enabled, there is a potential for data loss or corruption during a power failure.
- You can enable write cache only if a native operating system's file system—in Windows, the NTFS or FAT file system—resides on the array.
- If you do not want to enable the stripe set's array cache when you create it, you can do so later by using the `container set cache` command.

`/clear{=boolean}`

Specifies whether to set up the parity during RAID-5 creation by clearing the entire RAID-5 set. If you specify `TRUE`, the command sets up the parity during RAID-5 creation by clearing the RAID-5 set.

The default is `FALSE`; the command does not clear the entire RAID-5 set.

If you use neither this switch nor the `/scrub` switch, the command uses the scrub method by default.

Unlike the scrub method where the drive is immediately available, the clear method does not make the drive available for use until the parity-initialize operation completes.



If you specify `TRUE` for both the `/scrub` and `/clear` switches, the command displays an appropriate error message and returns to the prompt.

`/stripe_size{=integer}`

Specifies the stripe size for the RAID-5 set. Valid values are 16 KB, 32 KB, and 64 KB.

If you do not specify a value for this switch, it defaults to 64 KB.

`/label{=string}`

Specifies a label to be assigned to the newly created RAID-5 set. You can specify a maximum of sixteen characters for the label.

If you do not specify the switch, it defaults to no label. If you do not specify a label, you can do so later by using the `container set label` command.

Note that this label is not the label that displays in Windows Explorer. The label displayed by Windows Explorer comes from the label specified with the `container format` command.

`/scrub{=boolean}`

Specifies whether to set up the parity during RAID-5 creation by scrubbing the RAID-5 set. If you set this switch to `TRUE`, the command sets up the parity by scrubbing the RAID-5 set. Although the drive is immediately available, it is not parity-protected until the background scrub action completes.

This switch defaults to `TRUE`.

If you set this switch to `FALSE`, the command will set up parity by clearing the RAID-5 set. In other words, setting `/scrub=FALSE` enables the same behavior as `/clear=TRUE`.

This switch is supported on the following operating systems:

- Windows NT
- Windows 2000
- Windows XP
- NetWare

For UNIX, the RAID-5 set is always scrubbed.

`/wait{=boolean}`

Specifies whether the command prompt returns only after the parity-protect operation completes. If you set this switch to `TRUE`, the command prompt returns only after the RAID-5 set is parity-protected. The scrub or zero action completes.

The default is `FALSE`; the command prompt returns immediately before the parity-protect operation completes.

## Examples

The following example creates a RAID-5 set with the following characteristics:

- Creates the RAID-5 set on four disk drives connected to channel 0
- Specifies a freespace of 10 MB
- Uses the scrub method
- Indicates the command prompt return only after the scrub operation completes
- Specifies a stripe size of 64 KB
- Accepts the default label

```
AAC0>container create raid5 /stripe_size=64K /scrub
((0,02,0),10M) (0,03,0) (0,04,0) (0,05,0)

Executing: container create raid5 /stripe_size=65,536 /
scrub=TRUE ((CHANNEL=0,ID=2,LUN=0),10,485,760 )
(CHANNEL=0,ID=3,LUN=0) (CHANNEL=0,ID=4,LUN=0)
(CHANNEL=0,ID=5,LUN=0)

container 0 created
```

As the command executes, note that the title bar of the MS-DOS window displays the status of the command. For example:

```
Stat:OK!Task:101,Func:R5S Ctr:0,State:RUN 84.6%
```

See [page 3-10](#) for detailed information on the status information that the command displays in the title bar as it executes the task.

On UNIX systems, the message displayed after you execute the `container create raid5` command includes the root special file associated with the newly created RAID-5 set.

Use the `container list` command after using the `container create raid5` command to display information about the RAID-5 set.

```
AAC0>container list

Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
0	RAID-5	30.0MB	64KB	None	0:02:0 64.0KB: 10.0MB 0:03:0 64.0KB: 10.0MB 0:04:0 64.0KB: 10.0MB 0:04:0 64.0KB: 10.0MB

The following list describes the columns that contain information as a result of creating a RAID-5 set with the `container create raid5` command:

- The **Num Label** column displays the ID number of the newly created RAID-5 set, which in the example is 0.

If you specify a label when creating the RAID-5 set, it appears in this column. Because no label was specified when the RAID-5 set was created, no label appears in the column.

On UNIX systems, the root special file associated with the RAID-5 set also appears in this column.

- The **Type** column displays RAID-5 to indicate that the newly created array is a RAID-5 set.
- The **Total Size** column displays the total size of the freespace available. In this example, 40.0 MB of freespace was specified, which means that this column displays 30.0 MB as available.
- The **Stripe Size** column displays the stripe size you specified as the `stripe_size` parameter, which in the example is 64 KB.
- The **Usage** column displays **None** to indicate that the newly created RAID-5 set does not have a file system on it. To create an NTFS or FAT file system on an array, use the `container format` command.
- The **State** column displays the state of an array, which in the example shows **Unprot** to indicate that the RAID-5 set is not yet redundant. This means that the command has not completed the creation of the RAID-5 set. If the `container list` command is executed after the RAID-5 set completes execution, the State column is blank.

(Note that this column appears only if you specify the `/full` switch with the `container list` command.)

- The **Scsi C:ID:L** column displays the SCSI device ID(s) for the disk(s) on which you created the RAID-5 set, which in the example are: 0:02:0, 0:03:0, 0:04:0, and 0:05:0.
- The **Partition Offset: Size** column displays the partition offset(s) and size(s) of the newly created RAID-5 set, which in the example is 64.0 KB and 10.0 MB respectively.

## Related Information

container commands:

- `container list` ([page 5-54](#))
- `container set cache` ([page 5-97](#))
- `container set label` ([page 5-107](#))

## container create stripe

To create a stripe set from freespace, use the `container create stripe` command.

### Notes

If you created a stripe set you can run the `list devices` command on the NetWare console to verify its creation and then create the necessary NetWare disk partitions and volumes.

For information on how to create partitions and volumes, see the appropriate NetWare documentation.

### Syntax

```
container create stripe [/cache{=boolean}]
[/stripe_size{=integer}] [/label{=string}]
{free_space} [{scsi_device}...]
```

### Parameters

`{free_space}`

Specifies the SCSI device and its associated freespace used to create the stripe set. See the *free\_space* section on page 3-8 for more information on how to specify the freespace.

`{scsi_device}...`

Specifies one or more devices. A SCSI ID consists of a SCSI bus number (e.g., 0, 1, 2, 3, etc.), SCSI device ID (0 through 15 inclusive), and SCSI device logical unit number (0 through 7 inclusive). See the installation guide for your controller to determine the number of buses it actually supports.

See the *scsi\_device* section on page 3-7 for more information on how to specify the SCSI device. The command uses the freespace(s) from the SCSI device(s) to create the stripe set. The size from the device is identical to the one you specify in the *free\_space* parameter.

The minimum number of partitions for a stripe set is 2, and the maximum number is 16.

## Switches

`/cache{=boolean}`

When *FALSE* (the default), the command causes the controller to disable the array's array cache. *FALSE* means:

- Disables the read-ahead cache.
- Disables the write-back cache.

When *TRUE*, write cache is enabled, providing maximum performance.

### Notes

- When the write cache is enabled, there is a potential for data loss or corruption during a power failure.
- You can enable write cache only if a native operating system's file system—in Windows, the NTFS or FAT file system—resides on the array.
- If you do not want to enable the stripe set's array cache when you create it, you can do so later by using the `container set cache` command.

`/stripe_size{=integer}`

Specifies the stripe size for the stripe set. Valid values are 16 KB, 32 KB, and 64 KB.

If you do not specify a value for this switch, it defaults to 64 KB.

`/label{=string}`

Specifies a label to be assigned to the newly created stripe set. You can specify a maximum of sixteen characters for the label.

If you do not specify the switch, it defaults to no label. If you do not specify a label, you can do so later by using the `container set label` command.

Note that this label is not the label that displays in Windows Explorer. The label displayed by Windows Explorer comes from the label specified with the `container format` command.

## Examples

Before creating a stripe set, use the `disk show space` command to obtain information about the available devices, as in the following example:

```
AAC0>disk show space

Executing: disk show space
Scsi C:ID:L Usage      Size
-----
0:00:0   Free          64.0KB: 11.0MB
0:01:0   Free          64.0KB: 11.0MB
0:02:0   Free          64.0KB: 49.0MB
0:03:0   Free          64.0KB: 49.0MB
0:04:0   Free          64.0KB: 49.0MB
0:05:0   Free          64.0KB: 49.0MB
0:06:0   Free          64.0KB: 11.0MB
0:08:0   Free          64.0KB: 11.0MB
```

The example list shows that there are eight SCSI disks available to create the stripe set. There must be enough contiguous freespace on all the specified drives.

The following example shows how to create a stripe set with the following characteristics:

- 15 MB of freespace on the (0,2,0) SCSI disk
- The default stripe size of 64 KB
- Two freespaces of 15 MB on the (0,3,0) and (0,4,0) SCSI disks

```
AAC0>container create stripe ((0,2,0), 15M) (0,3,0) (0,4,0)

Executing: container create stripe ((CHANNEL=0,ID=2,LUN=0),15,728,640 )
(CHANNEL=0,ID=3,LUN=0) (CHANNEL=0,ID=4,LUN=0)
container 0 created
```

On UNIX systems, the message displayed after you execute the `container create stripe` command includes the root special file associated with the newly created stripe set.

Use the `container list` command after using the `container create stripe` command to display information about the stripe set, as in the following example:

```
AAC0>container list

Executing: container list
Num      Total  Oth Stripe      Scsi  Partition
Dr Label Type  Size   Ctr Size  Usage  C:ID:L Offset:Size
--
-----
```

```

0      Stripe 45.0MB      64KB  None    0:02:0 64.0KB: 15.0MB
                                0:03:0 64.0KB: 15.0MB
                                0:04:0 64.0KB: 15.0MB

```

The following list describes the items in the display that contain information as the result of creating a stripe set with the `container create stripe` command:

- The **Num Label** column displays the ID number of the newly created stripe set, which in the example is 0.  
  
If you specify a label when creating the stripe set, it appears in this column. Because no label was specified when the stripe set was created, no label appears in the column.  
  
On UNIX systems, the root special file associated with the array also appears in this column.
- The **Type** column displays the type `Stripe`, to indicate that the newly created array is a stripe set.
- The **Total Size** column displays the size of the freespace that makes up the stripe set. In this example, the total size consists of the freespace (15.0 MB) specified with SCSI device (0,2,0) and equal sizes associated with devices (0,3,0), and (0,4,0) for a total of 45.0 MB.
- The **Stripe Size** column displays the stripe size you specified for the stripe set. In this example, the stripe size is the default (32 KB).
- The **Usage** column displays **None** to indicate that the newly created stripe set does not have a file system on it. To create an NTFS or FAT file system on an array, use the `container format` command.



- The **Scsi C:ID:L** column displays the SCSI device ID(s) for the disk(s) whose associated freespaces you used to create the stripe set. In this example, the display shows 0:02:0, 0:03:0, and 0:04:0.
- The **Partition Offset: Size** column displays the partition offset(s) and the size(s) for the disk(s) whose associated freespaces you used to create the stripe set. In this example, the display shows 64.0KB:15MB, 64.0KB:15.0MB, and 64.0KB:15.0MB.

## Related Information

container commands:

- `container create mstripe` ([page 5-14](#))
- `container create mvolume` ([page 5-19](#))
- `container list` ([page 5-54](#))
- `container set cache` ([page 5-97](#))
- `container set label` ([page 5-107](#))

disk commands:

- `disk show space` ([page 7-22](#))

## container create volume

To create a volume set from freespace, use the `container create volume` command.

### Notes

If you created a volume set you can run the `list devices` command on the NetWare console to verify its creation and then create the necessary NetWare disk partitions and volumes.

For information on how to create partitions and volumes, see the appropriate NetWare documentation.

### Syntax

```
container create volume [/cache{=boolean}] [/
label{=string}] {scsi_device},{free_space}
[{scsi_device},{free_space}...]
```

### Parameters

`{scsi_device}`

Specifies the ID for the SCSI device whose freespace you want to use for creating the volume set. A SCSI ID consists of a SCSI bus number (e.g., 0, 1, 2, 3, etc.), SCSI device ID (0 through 15 inclusive), and SCSI device logical unit number (0 through 7 inclusive). See the installation guide for your controller to determine the number of buses it actually supports.

See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

Note that you must explicitly specify the entire ID for the SCSI device (the SCSI channel number, device ID, and device logical unit number).

`{free_space}`

Specifies the freespace used to create the volume set on the specified SCSI device. See the `free_space` section on page 3-8 for more information on how to specify the freespace.

```
{scsi_device},{free_space}...
```

Specifies any additional devices and associated freespaces used to create the volume set. A volume set can have a maximum of 16 partitions (or freespaces, since each freespace becomes a partition).

Note that you must explicitly specify the entire ID for any additional devices (the SCSI channel number, device ID, and device logical unit number).

## Switches

```
/cache{=boolean}
```

When *FALSE* (the default), the command causes the controller to disable the array's array cache. *FALSE* means:

- Disables the read-ahead cache.
- Disables the write-back cache.

When *TRUE*, write cache is enabled, providing maximum performance.

### Notes

- When the write cache is enabled, there is a potential for data loss or corruption during a power failure.
- You can enable write cache only if a native operating system's file system—in Windows, the NTFS or FAT file system—resides on the array.
- If you do not want to enable the stripe set's array cache when you create it, you can do so later by using the `container set cache` command.

```
/label{=string}
```

Specifies a label to be assigned to the newly created volume set. You can specify a maximum of sixteen characters for the label.

If you do not specify the switch, it defaults to no label. If you do not specify a label, you can do so later by using the `container set label` command.

Note that this label is not the label that displays in Windows Explorer. The label displayed by Windows Explorer comes

from the label specified with the `container format` command.

## Examples

Before creating a volume set, use the `disk show space` command to obtain information about the available devices, as in the following example:

```
AAC0>disk show space

Executing: disk show space

Scsi C:ID:L Usage      Size
-----
0:00:0  Free          64.0KB: 11.0MB
0:01:0  Free          64.0KB: 11.0MB
0:02:0  Free          64.0KB: 49.0MB
0:03:0  Free          64.0KB: 49.0MB
0:04:0  Free          64.0KB: 49.0MB
0:05:0  Free          64.0KB: 49.0MB
0:06:0  Free          64.0KB: 11.0MB
0:08:0  Free          64.0KB: 11.0MB
```

The example list shows that there are eight SCSI disks available to create the volume set.

The following example creates a volume set on SCSI disk (0,2,0) with a freespace of 15 MB:

```
AAC0>container create volume ((0,2,0), 15M)

Executing: container create volume ((CHANNEL=0,ID=2,LUN=0),15,728,640)
container 0 created
```

On UNIX systems, the message displayed after you execute the `container create volume` command includes the root special file associated with the newly created volume set.

Use the `container list` command after executing the `container create volume` command to display information about the volume set, as in the following example:

```
AAC0>container list

Executing: container list

Num      Total  Oth Stripe      Scsi  Partition
Dr Label Type  Size  Ctr Size  Usage  C:ID:L Offset:Size
-----
0        Volume 15.0MB          None  0:02:0 64.0KB: 15.0MB
```

The following list describes the columns that contain information as the result of creating a volume set with the `container create volume` command:

- The **Num Label** column displays the ID number of the newly created volume set, which in the example is 0.

If you specify a label when creating the volume set, it appears in this column. Because no label was specified when the volume set was created, no label appears in the column.

On UNIX systems, the root special file associated with the array also appears in this column.

- The **Type** column displays **Volume** to indicate that the newly created array is a volume set.
- The **Total Size** column displays the value(s) specified in the `free_space` parameter(s), which in the example is 15.0 MB.
- The **Usage** column displays **None** to indicate that the newly created volume set does not have a file system on it. To create an NTFS or FAT file system on an array, use the `container format` command.
- The **Scsi C:ID:L** column displays the SCSI device ID for the disk on which you created the volume set, which in the example is 0:02:0.
- The **Partition Offset: Size** column displays the partition offset and the size of the newly created volume set's partition, which in the example are 64.0 KB and 15.0 MB. The size is the size you specified in the `free_space` parameter(s).

## Related Information

container commands:

- `container extend volume` ([page 5-50](#))
- `container format` ([page 5-54](#))
- `container list` ([page 5-54](#))
- `container promote` ([page 5-69](#))
- `container set cache` ([page 5-97](#))
- `container set label` ([page 5-107](#))

disk commands:

- disk show space ([page 7-22](#))

## container delete

To delete an array from the currently opened controller, use the `container delete` command.

### Notes

If you are deleting an array on a NetWare system, you must first go to the NetWare console and remove the corresponding volume and partition. For information on how to perform these operations, see the appropriate NetWare documentation.

After you delete an array on a NetWare server, you must run the `list devices` command on the NetWare console so that the corresponding virtual disk is removed from the system's internal device table.

### Syntax

```
container delete [/always{=boolean}]  
[/unconditional{=boolean}]{container}
```

### Parameters

{container}

Specifies the ID number (0 to 63) of the array to delete. The array can be any array on the controller.

You can never delete an array if files are open on it.

for the syntax associated with specifying a partner and a container ID.

### Switches

/always{=boolean}

Specifies whether to delete the array, even if it has a file system. If you specify `TRUE`, the command deletes the array even if it has a file system. If you specify `FALSE`, the command deletes the array only if it has no file system.

This switch defaults to `FALSE`. In both cases, all user files must be closed; the `/always` switch cannot override this restriction.

```
/unconditional{=boolean}
```

Specifies whether to delete the array, even if the array has open files on it. If you specify `TRUE`, the command deletes the array even if it has open files on it.

The switch defaults to `FALSE` (that is, the container delete command does not delete an array that has open files on it).



**WARNING:** Unconditionally deleting an array that is in use can cause a system crash under some circumstances.

## Examples

Before deleting an array, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a multilevel volume set) on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume	40.0MB		NTFS	
63	Volume	10.0MB		0:02:0	64.0KB: 10.0MB
62	Volume	15.0MB		0:02:0	64.0KB: 10.0MB
61	Volume	15.0MB		0:04:0	64.0KB: 10.0MB

The following example shows how to delete array 0 using the `/always` switch to indicate a delete even though the array has a file system:

```
AAC0>container delete /always=TRUE 0
```

```
Executing: container delete /always=TRUE 0
```

Use the `container list` command after using the `container delete` command to confirm that the command actually deleted the specified array, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

```
No containers found.
```



Note that in the example, the deletion of array 0 caused the deletion of all the underlying (hidden) arrays (arrays 63, 62, and 61).

## **Related Information**

container commands:

- `container create mirror` ([page 5-5](#))
- `container create mmirror` ([page 5-9](#))
- `container create mstripe` ([page 5-14](#))
- `container create mvolume` ([page 5-19](#))
- `container create raid5` ([page 5-24](#))
- `container create stripe` ([page 5-30](#))
- `container create volume` ([page 5-35](#))
- `container list` ([page 5-54](#))

## container extend file\_system

To extend a file system so that it uses all of the space in an array, use the `container extend file_system` command. This command allows you to extend the NTFS file system.

Typically, you use this command after extending an array (by adding a level to it with the `container add_level` command and, possibly, by extending it with the `container extend mvolume` command).

If you extend an NTFS file system, you must reboot your system in order for the extension to take effect.

### Notes

The following notes relate to using the `container extend file_system` command to extend an NTFS file system:

- If you extend an NTFS file system, the command displays an appropriate message indicating that you need to reboot the system to show the new space. If you check the Windows Event Log, a message similar to the following appears:

```
The file system structure on the disk is corrupt and unusable. Please run the chkdsk utility on the device \Device\Harddisk0\Partition1 with label "".
```

The previous message always appears in the Windows Event Log even if the file system extend operation is successful. You do not need to run the `chkdsk` utility after a successful file system extend operation.

- You cannot extend an NTFS file system that resides on a boot array.

## Command And Switch Availability

This command and its associated switches (if any) are supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)

## Syntax

```
container extend file_system {container}
```

## Parameters

```
{container}
```

Specifies the ID number (0 to 63) of the array whose file system you want to extend.

## Examples

Typically, you would use the `container extend file_system` command after adding a level to an array and, perhaps, after extending a multilevel volume set.

Before extending a file system, use the `container list` command to obtain information about any existing arrays:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type Size	Ctr	Size	Usage	C:ID:L	Offset:Size
E: 0	Volume	300MB			
63	Volume	100MB			
62	Volume	100MB			
61	Volume	100MB			

The following list describes the sequence that caused the previous example to display:

- 1 Create a volume set of 100 MB on disk (1,0,0) with the `container create volume` command.
- 2 Add a level to the volume set created in Step 1 with the `container add_level` command.

- 3 Assign a drive letter to the volume set created in Step 1 with the `container assign drive_letter` command.
- 4 Create an NTFS file system on the volume set created in Step 1 with the `container format` command.
- 5 Create a second volume set of 100 MB on disk (1,1,0) with the `container create volume` command.
- 6 Create a third volume set of 100 MB on disk (1,2,0) with the `container create volume` command.
- 7 Used the `container extend mvolume` command to add the arrays created in Steps 5 and 6 to the array created in Step 1.

The following example extends the NTFS file system to make use of the additional arrays:

```
AAC0>container extend file_system 0
```

```
Executing: container extend file_system 0
```

The system has been marked to expand the file system on the next reboot.

Reboot the system to get the file system to expand to show the new space.

Use the `container list` command after using the `container extend file_system` command to display information about the array after you extend a file system. Note that there is no change in the display as a result of using the `container extend file_system` command.

## Related Information

container commands:

- `container format` ([page 5-54](#))

## container extend mvolume

To extend a multilevel volume set by adding one or more arrays to it, use the `container extend mvolume` command. Any file system on the multilevel volume set remains intact, and can be extended to include the added space.

### Command And Switch Availability

This command and its associated switches (if any) are supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)

### Syntax

```
container extend mvolume {container_to_extend}  
{container} [{container}...]
```

### Parameters

`{container_to_extend}`

Specifies the ID number (0 to 63) of the array to extend. Typically, this array is a multilevel volume set.

Specifies the ID number (0 to 63) of the array to add to the previously specified multilevel volume set. If a file system exists on this array, the command displays an appropriate error message and does not allow you to create the multilevel volume set. This prevents the loss of any data residing in files on the array.

{container}...

Specifies the ID number (0 to 63) of the array or arrays to add to the previously specified multilevel volume set. The previous file system information applies to these additional arrays as well.

## Examples

Before extending a multilevel volume set, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a multilevel volume set) on this controller. In addition, the example shows array 1 and array 2 (both volume sets created with the `container create volume` command).

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label	Type	Size	Ctr Size	Usage	C:ID:L Offset:Size
F: 0	Volume	10MB			
63	Volume	10MB			
1	Volume	15MB		None	0:03:0 64.0KB: 15.0MB
2	Volume	15MB		None	0:04:0 64.0KB: 15.0MB

The following example extends a multilevel volume set (array 0) by adding arrays 1 and 2 (which are both volume sets):

```
AAC0>container extend mvolume 0 1 2
```

```
Executing: container extend mvolume 0 1 2
```

```
container 0 extended
```

As the command executes, note that the title bar of the DOS window displays the status of the command. For example:

```
Stat:OK!Task:102,Func:FSX Ctr:0,State:RUN 84.3%
```

See [page 3-10](#) for detailed information on the status information that the command displays in the title bar as it executes the task.

Use the `container list` command after using the `container extend mvolume` command to display information about the

multilevel array of volume sets after you extend it, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth Stripe	Scsi	Partition
Dr Label Type Size Ctr Size Usage C:ID:L Offset:Size				
-----				
F: 0	Volume 40MB		NTFS	
63	Volume 10MB		0:02:0	64.0KB: 10.0MB
62	Volume 15MB		0:03:0	64.0KB: 15.0MB
61	Volume 15MB		0:04:0	64.0KB: 15.0MB

The following list describes the change to the display as the result of extending a multilevel array with the `container extend mvolume` command:

- The **Num Label** column continues to display the ID number (0) of the volume set previously created with the `container create volume` command and the ID number (63) of the volume set previously created with the `container add_level` command. In addition, the column displays the ID numbers (in this example, 62 and 61) of the volume sets created as a result of the `container extend mvolume` command.

If you specified labels when creating the volume sets, they appear in this column. Because no labels were specified when the volume sets were created, no labels appear in the column.

- The **Type** column continues to display Volume to indicate that the arrays previously created with the `container create volume` and the `container add_level` commands are volume sets. In addition, the column displays Volume for the volume sets created as a result of the `container extend mvolume` command.
- The **Total Size** column displays a new total size, taking into account the freespaces associated with the volume sets created as a result of the `container create volume` and the `container add_level` commands.
- The **Scsi C:ID:L** column displays the SCSI IDs for the devices on which the volume sets were created, which in the example are 0:02:0, 0:03:0, and 0:04:0.

- The **Partition Offset:Size** column displays the offsets and sizes for the partition(s) on which the volume sets were created, which in the example are 64.0KB:10MB, 64.0KB:15.0MB, and 64KB:15MB.

If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column changes to a “!” (exclamation point). See the `disk remove dead_partitions` ([page 7-9](#)) command for more information on dead partitions.

## Related Information

container commands:

- `container add_level` ([page 5-3](#))
- `container create mvolume` ([page 5-19](#))
- `container create volume` ([page 5-35](#))
- `container extend file_system` ([page 5-43](#))
- `container list` ([page 5-54](#))

disk commands:

- `disk remove dead_partitions` ([page 7-9](#))



## container extend volume

To extend a volume set by adding freespace to it, use the `container extend volume` command. The file system on the volume set remains intact and can be extended to include the added space.

### Command And Switch Availability

This command and its associated switches (if any) are supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)

### Syntax

```
container extend volume {container}
[ {free_space} ] [ {free_space} ... ]
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array (volume set) to extend.

`{free_space}`

Specifies the SCSI device and its associated freespace used to extend the specified array (volume set). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device. See the `free_space` section on page 3-8 for more information on how to specify the freespace.

`{free_space}...`

Specifies a SCSI device or devices and associated freespace or freespaces used to extend the specified array (volume set). You can specify a maximum of 16 freespace elements. If you exceed 16 freespace elements, the command displays an appropriate error message.

## Examples

Before extending a volume set, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array (volume set) 1 on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
E: 1	Volume	100MB		NTFS	2:02:0 200MB: 100MB

Before extending the volume set, you might want to use the `disk show space` command to display space usage information on the devices from which you plan to add freespace. The following example extends a volume set by adding freespace from two devices:

```
AAC0>container extend volume 1 ((2,0,0), 32M) ((2,1,0), 32M)
```

```
Executing: container extend volume 1
```

```
((CHANNEL=2,ID=0,LUN=0),33,554,432)((CHANNEL=2,ID=1,LUN=0),33,554,432)
```

```
container 1 expanded.
```

As the command executes, note that the title bar of the DOS window displays the status of the command. For example:

```
Stat:OK!,Task:101,Func:FSX Ctr:0,State:RUN 83.8%
```

See [page 3-10](#) for detailed information on the status information that the command displays in the title bar as it executes the task.

Use the `container list` command after using the `container extend volume` command to display information about the volume set after you extend it, as in the following example:

```
AAC0>container list

Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition			
Dr	Label	Type	Size	Ctr	Size	Usage	C:ID:L	Offset:Size
E: 1		Volume	164MB			NTFS	2:02:0	200MB: 100MB
							2:00:0	64.0KB: 32.0MB
							2:01:0	64.0KB: 32.0MB

The following list describes the change to the display as the result of extending a volume set with the `container extend volume` command:

- The **Total Size** column displays a new size, taking into account the value(s) specified for the `free_space` parameters. In the example the new size is 164 MB.
- The **Usage** column continues to display NTFS, indicating that array 1 has an NTFS file system on it.
- The **Scsi C:ID:L** column displays the SCSI ID for the disk on which the original volume set was created. It also displays the SCSI ID(s) for the disk(s) from which you specified the freespace. In the example, these disks are 2:00:0 and 2:01:0.
- The **Partition Offset:Size** column displays the partition offset(s) and size(s) for the partition(s) associated with the extended volume set. In the example the partition offsets and sizes are 64.0 KB and 32.0 MB.

If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column changes to a “!” (exclamation point). See the `disk remove dead_partitions` ([page 7-9](#)) command for more information on dead partitions.

## **Related Information**

container commands:

- container create volume ([page 5-35](#))
- container list ([page 5-54](#))

disk commands:

- disk remove dead\_partitions ([page 7-9](#))
- disk show space ([page 7-22](#))

## container list

To display information about one or all arrays on the controller, use the `container list` command. The display contains such information as the container's ID number and other useful information. Typically, you use the `container list` command to obtain specific information about arrays prior to using other array-related commands.

### Syntax

```
container list [/all{=boolean}] [/full{=boolean}]  
[ {container} ]
```

### Parameters

{container}

Specifies the ID number (0 to 63) of the array whose information you want to display. To display information about all of the arrays on the system, omit the ID number from the command.

### Switches

/all{=boolean}

Specifies whether to list all arrays on the system. If you specify `TRUE`, the command displays all arrays on the system. If you specify `FALSE`, the command displays only the array you specify in the command.

This switch defaults to `FALSE` if you specify a container. Otherwise, the switch defaults to `TRUE`.

/full{=boolean}

Specifies whether to display detailed information. If you specify `TRUE`, the command displays detailed information. If you specify `FALSE`, the command does not display detailed information.

This switch defaults to `FALSE`.

## Examples

The following example shows how to list nondetailed information about all arrays on the system:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type Size Ctr Size Usage C:ID:L Offset:Size					
0	Volume	15.0MB		None	0:02:0 64.0KB: 15.0MB
1	Stripe	45.0MB	32KB	None	0:02:0 15.0MB: 15.0MB
					0:03:0 64.0KB: 15.0MB
					0:04:0 64.0KB: 15.0MB

The detailed container list display contains the previous columns of information plus the following columns:

- State
- RO
- Lk
- Task
- Done%
- Ent
- Creation Date
- Creation Time
- Boot Device
- System Files

The following sections provide brief descriptions of each container list column.

### The Dr Column

On Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only) systems, this column displays a blank space if you did not assign a drive letter to the array. Otherwise, the column contains the letter associated with the array. Use the `container assign drive_letter` command to assign a drive letter to an array.

The **Dr** column does not appear on UNIX or NetWare systems.

### The Num Label Column

This column displays the container ID (a number from 0 to 63 inclusive). Typically, the CLI `container create`-related commands automatically assign an ID to an array. However, you can renumber an array with the `container move` command.

This column also displays the label assigned to the array when the array was created. If no label was assigned to the array then no label appears in the column. You can assign labels with the following commands:

- `container create mstripe`
- `container create mvolume`
- `container create raid5`
- `container create stripe`
- `container create volume`
- `container set label`

If your controller is running on a UNIX operating system, then the **Num Label** column also displays a root special file. On UNIX systems, arrays are associated with root special files not drive letters.

See your UNIX documentation for information on how to mount the array or to create a file system.

### The Type Column

This column displays the type of array. Table 5-1 describes the type values that can display in the Type column.

Table 5-1. Container Types

container Type	Meaning
Mirror	The container is a mirror set.
Stripe	The container is a stripe set.
Volume	The container is a volume set.
RAID-5	The container is a RAID-5 set.
Reconf	The container was reconfigured.

The CLI automatically assigns the array type as a result of creating arrays with the `container create`-related commands.

The CLI assigns the array type `Reconf` when the array is reconfigured from one array type to another. The `container reconfigure` command provides switches that allow you to reconfigure an array into specific array types.

The `Reconf` array type does not appear on UNIX systems.

### The Total Size Column

This column displays the size of the array. You specify this size when creating the array with one of the `container create`-related commands.



### The Stripe Size Column

This column displays the stripe size for the array. You specify the stripe size when using the `container create mstripe`, `container create raid5`, and `container create stripe` commands.

### The Usage Column

This column displays information about the data on or the status of the array. Specifically, the column can display the items listed in Table 5-2.

Table 5-2. Container Usage Items

Item	Meaning
FAT	The FAT file system resides on this array.
MultPart	A file system resides on a multi-partition array. A multi-partition array is an array that has multiple operating system or DOS partitions.
NetWare	The array resides in a NetWare environment.
None	No file system resides on this array.
NTFS	The NTFS file system resides on this array.
Open	For UNIX, there is a mounted file system on this array.
UnCfged	The array is a phantom array or the array is offline.
Unknown	An unknown file system resides on this array. For UNIX, the operating system recognized this array, and there has not been a query (e.g., mount, fdisk, read, write) on the array.
UnMap'd	The array is unusable and cannot be mounted.
Valid	The UNIX operating system recognized this array, and there has been a query (e.g., mount, fdisk, read, write) on the array.

### The Scsi C:ID:L Column

This column displays the SCSI channel number, the SCSI ID, and the SCSI logical unit number for the SCSI disk or disks on which the array was created.

### The Partition Offset:Size Column

This column displays the offset and size for the underlying partitions.

If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column changes to a “!” (exclamation point). See the `disk remove dead_partitions` (page 7-9) command for more information on dead partitions.

### The State Column

This column displays information about the state of the array. Specifically, the column can display the items listed in Table 5-3.

Table 5-3. Container State Items

Container State	Meaning
Copy	Indicates that the array is the copy array in an array reconfiguration operation.
Create	Indicates the creation of a mirror set.
Dest	Indicates that the array is the destination array in an array reconfiguration operation.
Normal	Indicates that the mirror set is in the normal state.
Raid5	Indicates that the array is a RAID-5 set in an array reconfiguration operation.
Source	Indicates that the array is the source array in an array reconfiguration operation.
Temp	Indicates that the array is a temporary array in an array reconfiguration operation.
Unprot	The RAID-5 set is not redundant. The <code>Unprot</code> array state typically displays during the creation of the RAID-5 set. Upon completion of RAID-5 set creation, the CLI replaces the <code>Unprot</code> state with the <code>Normal</code> state.

### The RO Column

This column displays the letters `RO` if the array is read-only. Otherwise, if the array is read-write the column displays a blank space. You can explicitly make an array read-only and read-write

with the `container readonly` and `container readwrite` commands.

Some CLI commands cause an array to temporarily become read-only without the use of the `container readonly` command.

### The Lk Column

This column displays the letter `L` if the array is locked. Otherwise, if the array is not locked, the column displays a blank space. You can explicitly lock and unlock an array with the `container lock` and `container unlock` commands.

Some CLI commands (during their execution) cause the array to become locked. Typically these commands unlock the array upon completing execution.

### The Task Column

This column displays the task or tasks running on a specified array or SCSI disk. Specifically, the column can display the items listed in Table 5-4.

Table 5-4. Task Types

Task	Meaning
Create	A create mirror set or create multilevel mirror set task is running on the specified array. When the create mirror set or create multilevel mirror set task completes, the specified array is a mirror set or a multilevel array of mirror sets. The create a mirror set or create a multilevel mirror set task runs as a result of using the <code>container create mirror</code> or <code>container create mmirror</code> command.
FmtFAT	An array format FAT file system task is running on the specified array. The format FAT file system task runs as a result of using the <code>container format</code> command with the <code>/file_system</code> switch set to FAT.
FmtNTFS	An array format NTFS file system task is running on the specified array. The format NTFS file system task runs as a result of using the <code>container format</code> command with the <code>/file_system</code> switch set to NTFS.

Table 5-4. Task Types (Continued)

Task	Meaning
Rebuild	A rebuild task is running on the specified array. Typically, the rebuild task runs when the controller is in a rebuild for a redundant array (RAID-5 set, mirror set, or multilevel array of mirror sets).
Reconfg	An array reconfigure task is running on the specified array. When the array reconfigure task completes, the specified array becomes a different array (for example, from a volume set to a stripe set). The array reconfigure task runs as a result of using the <code>container reconfigure</code> command.
Scrub	A scrub task is running on the specified redundant array. When the scrub task completes, the specified redundant array has reconstructed data on one partition based on data found on the other partition (for mirror sets and multilevel arrays of mirror sets). Or, the specified redundant array recalculates and replaces, if necessary, the parity information (for RAID-5 sets). The scrub task runs as a result of the <code>container scrub</code> command. The scrub task also runs as a result of the <code>container create raid5</code> command with the <code>/scrub</code> switch specified.
Verify	A verify with no repair of bad blocks task is running on the specified SCSI disk. When the verify with no repair of bad blocks task completes, the specified SCSI disk's blocks were verified without repairing any detected defects. The verify with no repair of bad blocks task runs as a result of using the <code>disk verify</code> command without specifying the <code>/repair</code> switch.
VfyRepl	A verify with repair of bad blocks task is running on the specified SCSI disk. When the verify with repair of bad blocks task completes, the specified SCSI disk's blocks are verified with repairs. The verify with repair of bad blocks task runs as a result of using the <code>disk verify</code> command with the <code>/repair</code> switch.
Zero	A clear disk task is running on the specified SCSI disk. When the clear disk task completes, the specified SCSI disk is cleared (that is, all data is erased and cannot be recovered). The clear disk task runs as a result of using the <code>disk zero</code> command.

### The Done % Column

This column displays a running percentage value that indicates the progress of the currently running task. The display shows the percentage in tenths of a percent increments. The currently running task is complete when the Done % column displays 100%.

### The Ent Column

This column displays the number of elements associated with each array. The maximum is 16 elements.

### The Creation Date and Creation Time Columns

The Creation Date column displays the date on which you created the array. The following example shows an example of a date that can appear in this column:

082999

As the example shows, the command displays the date in the form `mmddyy` where

- `mm` is a two-digit number that indicates the month in which you created the array. For example, 01 indicates the month of January, 12 indicates the month of December, and so forth.
- `dd` is a two-digit number that indicates the day of the month in which you created the array. For example, 29 indicates the 29th day.
- `yy` indicates the year you created the array. For example, 97 indicates the array was created in the year 1997.

The **Creation Time** column displays the time in 24-hour format at which you created the array. The following example shows an example of a time that can appear in this column:

11:55:49

As the example shows, the command displays the time in the form `hhmmss` where

- `hh` is a two-digit number that indicates the hour at which you created the array.
- `mm` is a two-digit number that indicates the minute at which you created the array.

- `ss` is a two-digit number that indicates the second at which you created the array.

### **The Boot Device Column**

This column displays a value that indicates whether the array resides on the boot device. An X appears in the column if the array resides on the boot device. Otherwise, if the array does not reside on the boot device, no value appears in the column.

The **Boot Device** column does not appear on UNIX systems.

### **The System Files Column**

This column displays a value that indicates whether the array resides on the system device. An X appears in the column if the array resides on the system device. Otherwise, if the array does not reside on the system device, no value appears in the column.

## **Related Information**

container commands:

- container create mirror ([page 5-5](#))
- container create mstripe ([page 5-14](#))
- container create raid5 ([page 5-24](#))
- container create stripe ([page 5-30](#))
- container scrub ([page 5-94](#))

disk commands:

- disk verify ([page 7-25](#))
- disk zero ([page 7-27](#))

## container lock



**Caution:** Use the `container lock` command only under the direction of technical support.

To lock an array into volatile memory space on the currently opened controller, use the `container lock` command. When an array is locked into volatile memory space, the `container reconfigure` command has no effect. A locked array cannot be moved, deleted, made read-only, or used to create a multilevel array.

### Syntax

```
container lock {container}
```

### Parameters

```
{container}
```

Specifies the ID number (0 to 63) of the array to lock into volatile memory space.

### Examples

Before locking an array, use the `container list` command (with the `/full` switch) to obtain information about any existing arrays. As the following example shows, there is an array 0 (a volume set) on this controller. Note that the **Lk** column is blank:

```
AAC0>container list /full
```

```
Executing: container list /full=TRUE
```

Num	Total	Oth	Stripe	Scsi	
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L
-----					
F: 0	Volume	10.0MB		NTFS	0:02:0

Note that the example eliminates some items in the `container list` display so that you can see an example of the **Lk** column.

The following example locks array 0 into volatile memory space:

```
AAC0>container lock 0
```

```
Executing: container lock 0
```



Use the `container list` command (with the `/full` switch) after using the `container lock` command to display information about the array you just locked, as in the following example:

```
AAC0>container list /full

Executing: container list /full=TRUE
  Num      Total  Oth Stripe      Scsi
Dr Label Type  Size  Ctr Size  Usage  C:ID:L  Lk
--  ----  -
F: 0      Volume 10.0MB      NTFS    0:02:0  L
```

Note that the example eliminates some items in the `container list` display so that you can see an example of the **Lk** column.

As the result of locking an array (in this example, array 0) with the `container lock` command, the **Lk** column displays **L** (instead of a blank) to indicate that the specified array is now locked.

## Related Information

container commands:

- `container list` ([page 5-54](#))
- `container unlock` ([page 5-117](#))

## container move

To renumber an array, use the `container move` command. If another array already has the new number, the command returns an error.

### Notes

A controller currently supports 24 visible arrays. A visible array is an array that is visible to the operating system and users. Visible arrays are identified with container IDs 0 through 23.

Container IDs 24 through 63 are reserved for hidden arrays. A hidden array is an array that is not visible to the operating system and can only be used by other arrays. (The `container list` command, however, displays hidden arrays.)

The `container move` command does not prevent you from assigning a hidden container ID (24 through 63) to a visible array. If you assign a visible array with a hidden container ID (24 through 63), the operating system no longer sees the visible array. In general, when renumbering a visible array, use container IDs 0 through 23. When renumbering a hidden array, use container IDs 24 through 63.

### Syntax

```
container move {container} {container}
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array to renumber.

`{container}`

Specifies the ID number (0 to 63) to assign to the array specified in the first `container` parameter.

## Examples

Before renumbering an array, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a volume set) on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume	20.0MB		NTFS	0:02:0 64.0KB: 10.0MB
					0:03:0 64.0KB: 10.0MB

The following example renumbers array 0 to array 5:

```
AAC0>container move 0 5
```

```
Executing: container move 0 5
```

Use the `container list` command after using the `container move` command to display information about the array you just renumbered, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 5	Volume	20.0MB		NTFS	0:02:0 64.0KB: 10.0MB
					0:03:0 64.0KB: 10.0MB

As a result of renumbering array 0 to array 5 with the `container move` command, the **Num Label** column displays ID number 5 instead of 0. This column also displays the label assigned to the array when the array was created. If no label was assigned to the array then no label appears in the column.

## Related Information

container commands:

- `container list` ([page 5-54](#))

## container promote

To create a multilevel volume set from a stripe set or volume set, use the `container promote` command. In this case, the resulting array is an array of single-partition volume sets, each of which has a single partition from the original array.

Typically, you use the `container promote` command as part of a process to provide fault tolerance for existing volume sets. After `container promote` is executed, use `container create mmirror` (which provides the fault tolerance) to make this array an array of mirror sets.

### Notes

The `container promote` command differs from the `container add_level` command as follows:

- The `container create volume` command creates a single-level array.
- If you use the `container add_level` command and specify the ID for a single-level volume set, the result is a volume set with two partitions.
- If you use the `container promote` command and specify the ID for a single-level volume set, the result is a multilevel volume set (a volume set with two volumes under it).

### Syntax

```
container promote {container}
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array to promote into a multilevel array.

## Examples

Before promoting an array to a multilevel array, use the `container list` command to obtain information about any existing arrays.

As the following example shows, there is an array 0 (a stripe set) on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num		Total	Oth	Stripe		Scsi	Partition
Dr	Label	Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
0		Stripe	45.0MB		32KB	None	0:02:0 64.0KB: 15.0MB
							0:03:0 64.0KB: 15.0MB
							0:04:0 64.0KB: 15.0MB

The following example promotes array 0 to a multilevel array:

```
AAC0>container promote 0
```

```
Executing: container promote 0
```

Use the `container list` command after using the `container promote` command to display information about the array you just promoted, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num		Total	Oth	Stripe		Scsi	Partition
Dr	Label	Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
0		Stripe	45.0MB		32KB	None	
63		Volume	15.0MB				0:02:0 64.0KB: 15.0MB
62		Volume	15.0MB				0:03:0 64.0KB: 15.0MB
61		Volume	15.0MB				0:04:0 64.0KB: 15.0MB

The following list describes the change to the display as the result of promoting an array (in this example, array 0) with the `container promote` command:

- The **Num Label** column displays four container IDs: 0 (the array that was promoted to a multilevel volume set), 63 (one of the underlying volume sets), 62 (a second underlying volume set), and 61 (a third underlying volume set).

This column also displays the label assigned to the array when the array was created. If no label was assigned to the array then no label appears in the column.

- The **Type** column displays the Volume array type for the three underlying volume sets.
- The **Total Size** column displays the sizes of the arrays. Note that the underlying arrays are each 15.0 MB and array 0's size is the total of the two underlying arrays.
- The **Scsi C:ID:L** column displays the SCSI channel number, SCSI device ID, and SCSI logical unit number for the underlying arrays. In the example, array 63 resides on disk 0:02:0, array 62 resides on disk 0:03:0, and array 61 resides on disk 0:04:0.
- The **Partition Offset:Size** column displays the offset and size for the underlying arrays' partitions. In the example, arrays 63, 62, and 61 have 64.0 KB offsets and 15.0 MB sizes.

If a partition is dead, the ":" (colon) in the **Partition Offset:Size** column changes to a "!" (exclamation point). See the `disk remove dead_partitions` (page 7-9) command for more information on dead partitions.

As stated previously, you can now use the `container create mmirror` command to make a promoted array an array of mirror sets.

## Related Information

container commands:

- `container create mmirror` (page 5-9)
- `container create volume` (page 5-35)
- `container list` (page 5-54)

disk commands:

- `disk remove dead_partitions` (page 7-9)

## container readonly

To change an array's read-write status to read-only status, use the `container readonly` command. Subsequent modifications to the data on the array are prohibited. To use the `container readonly` command, the array cannot be in use by any application.

### Syntax

```
container readonly {container}
```

### Parameters

```
{container}
```

Specifies the ID number (0 to 63) of the array to make read-only.

### Examples

Before making an array read-only, use the `container list` command (with the `/full` switch) to obtain information about any existing arrays. As the following example shows, there is an array 0 (a volume set) on this controller. Note that the **RO** column is blank:

```
AAC0>container list /full
```

```
Executing: container list /full=TRUE 0
```

Num	Total	Oth	Stripe	Scsi
Dr Label Type	Size	Ctr	Size	Usage
C:ID:L	RO			
F: 0	Volume 10.0MB			NTFS
				0:02:0

Note that the example eliminates some items in the `container list` display so that you can see an example of the **RO** column.

The following example changes array 0 from read-write status to read-only status:

```
AAC0>container readonly 0
```

```
Executing: container readonly 0
```

Use the `container list` command (with the `/full` switch) after using the `container readonly` command to display information about the array you just made read-only, as in the following example:

```
AAC0>container list /full
```

```
Executing: container list /full=TRUE
```

Num	Total	Oth	Stripe	Scsi
Dr Label Type	Size	Ctr	Size	Usage
				C:ID:L
F: 0	Volume 10.0MB			NTFS 0:02:0
				RO

Note that the example eliminates some items in the `container list` display so that you can see an example of the **RO** column.

As a result of making an array (in this example, array 0) read-only with the `container readonly` command, the **RO** column displays RO (instead of a blank) to indicate that the specified array is now read-only.

## Related Information

container commands:

- `container list` ([page 5-54](#))
- `container readwrite` ([page 5-74](#))



## container readwrite

To change an array's read-only status to read-write status, use the `container readwrite` command. Subsequent modifications to the data on the array are allowed. To use the `container readwrite` command, none of the array's files can be open.

### Syntax

```
container readwrite {container}
```

### Parameters

```
{container}
```

Specifies the ID number (0 to 63) of the array to make read-write.

### Examples

Before making an array read-write, use the `container list` command (with the `/full` switch) to obtain information about any existing arrays. As the following example shows, there is an array 0 (a volume set) on this controller. Note that the **RO** column displays the value **RO**:

```
AAC0>container list /full
```

```
Executing: container list /full=TRUE
```

Num	Total	Oth	Stripe	Scsi	
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L RO
-----					
F: 0	Volume	10.0MB		NTFS	0:02:0 RO

Note that the example eliminates some items in the `container list` display so that you can see an example of the **RO** column.

The following example changes array 0 from read-only status to read-write status:

```
AAC0>container readwrite 0
```

```
Executing: container readwrite 0
```

Use the `container list` command (with the `/full` switch) after using the `container readwrite` command to display information about the array you just made read-write, as in the following example:

```
AAC0>container list /full

Executing: container list /full=TRUE
```

Num	Total	Oth	Stripe	Scsi	
Dr Label	Type	Size	Ctr Size	Usage	
C:ID:L	RO				
F: 0	Volume	10.0MB		NTFS	0:02:0

Note that the example eliminates some items in the `container list` display so that you can see an example of the **RO** column.

As a result of making an array (in this example, array 0) read-write with the `container readwrite` command, the **RO** column displays blank (instead of an RO) to indicate that the specified array is now read-write.

## Related Information

container commands:

- `container list` ([page 5-54](#))
- `container readonly` ([page 5-72](#))

## container reconfigure

To change the configuration of an array, use the `container reconfigure` command. The `container reconfigure` command allows you to perform the following operations:

- Transform an array from one type to another type

You can transform an array from one type to another type. For example, you can transform a stripe set into a RAID-5 set. In this case, the stripe set changes to a RAID-5 set and remains the same size. You specify the appropriate switches that the `container reconfigure` command provides to transform an array.

- Add more SCSI drives to an array

You can add more SCSI drives to an array by specifying one or more devices. This action extends the array. The command uses the freespace(s) from the SCSI device(s) you specify to reconfigure the array.

For example, you can extend a stripe set by specifying one or more devices. In this case, the stripe set grows, but remains a stripe set.

- Change an array's stripe size

You can change an array's stripe size by specifying the appropriate switch with a valid stripe size. For example, you can change the stripe size of a stripe set from 16K to 32K.

- Change an array's partition size

You can change an array's partition size by specifying the appropriate switch with a valid partition size. For example, you can change the partition size of a mirror set from 500 MB to 800 MB.

- **Extend an existing file system**

You can extend an existing file system by specifying the appropriate switch. (The command extends the file system after completing the reconfigure operation.)

- **Move partitions to other disks**

You can move an array's partitions to other disks by specifying the appropriate switches with a valid partition or partitions. For example, you can move three entries of a RAID-5 set to new disks.

You can combine the previously listed operations.

## **Notes**

Some reconfigure operations result in destination sizes slightly different than what you specified.

When you perform an array reconfigure operation on a UNIX operating system you may see a resource conflict-related error. Typically, this error occurs when there is not enough space on the disk to perform the reconfigure operation. To correct the problem, use another disk (with more space) to perform the reconfigure operation.

## **Syntax**

```
container reconfigure [/stripe_size{=integer}]  
[/extend_fs{=boolean}][[/mirror{=boolean}]]  
[/partition_move{=boolean}]]  
[/partition_size{=integer}]] [/raid5{=boolean}]]  
[/raid10{=boolean}][[/volume{=boolean}]]  
[/wait{=boolean}]] {container} [{scsi_device}...]
```

## Parameters

`{container}`

Specifies the ID number (0 to 63) of the array you want to reconfigure.

`{scsi_device}...`

Specifies one or more devices. Typically, you specify one or more devices when adding more drives to an array or moving an array's partitions. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), a SCSI device ID (0 through 15 inclusive), and a SCSI device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device. The command uses the freespace(s) from the SCSI device(s) you specify to reconfigure the array.

## Switches

`/stripe_size{=integer}`

Specifies the changed stripe size for the array. You can specify stripe sizes for the following types of arrays:

- RAID-5 set
- Multilevel stripe set:
  - Stripe set of mirror sets
  - Stripe set of volume sets
  - Stripe set of stripe sets

`/extend_fs{=boolean}`

Specifies whether to extend the file system so that it occupies the entire reconfigured array. If you specify `TRUE`, the command adds no new freespace and extends the file system so that it occupies the entire reconfigured array. (The command extends the file system after completing the reconfigure operation.) If you specify `FALSE`, the command adds freespace to the reconfigured array and does not extend the file system to occupy the entire reconfigured array.

This switch defaults to `FALSE`.

This switch applies only to those multilevel volume sets on which an NTFS file system resides. If you extend an NTFS file system, you must reboot your system in order for the extension to take effect.

If you do not specify this switch, you can later extend the file system with the `container extend file_system` command.

This command and its associated switches (if any) are supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)

`/mirror{=boolean}`

Specifies whether to reconfigure the existing array into a mirror set. If you specify `TRUE`, the command reconfigures the existing array into a mirror set. If you specify `FALSE`, the command does not reconfigure the existing array into a mirror set.

This switch defaults to `FALSE`.

`/partition_move{=boolean}`

Specifies whether to move partitions instead of adding extra space (using additional disks). If you specify `TRUE`, the command moves partitions instead of adding extra space. If you specify `FALSE`, the command does not move partitions.

This switch defaults to `FALSE`.

`/partition_size{=integer}`

Specifies the partition size. All partitions must be the same size and you can specify only one partition per disk.

`/raid5{=boolean}`

Specifies whether to reconfigure the existing array into a RAID-5 set. If you specify `TRUE`, the command reconfigures the existing array into a RAID-5 set. If you specify `FALSE`, the command does not reconfigure the existing array into a RAID-5 set.

This switch defaults to `FALSE`.

`/raid10{=boolean}`

Specifies whether to reconfigure the existing array into a RAID-10 set. If you specify `TRUE`, the command reconfigures the existing array into a RAID-10 set. If you specify `FALSE`, the command does not reconfigure the existing array into a RAID-10 set.

This switch defaults to `FALSE`.

`/stripe{=boolean}`

Specifies whether to reconfigure the existing array into a stripe set. If you specify `TRUE`, the command reconfigures the existing array into a stripe set. If you specify `FALSE`, the command does not reconfigure the existing array into a stripe set.

This switch defaults to `FALSE`.

`/volume{=boolean}`

Specifies whether to reconfigure the existing array into a volume set. If you specify `TRUE`, the command reconfigures the existing array into a volume set. If you specify `FALSE`, the command does not reconfigure the existing array into a volume set.

This switch defaults to `FALSE`.

```
/wait{=boolean}
```

Specifies whether the command performs the array reconfiguration operation synchronously or asynchronously. If you set this switch to `TRUE`, the command performs the array reconfiguration operation synchronously, which means the command prompt does not return until the array reconfiguration operation completes. If you set this switch to `FALSE`, the command performs the array reconfiguration operation asynchronously, which means the command prompt returns immediately.

## Examples

Use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a mirror set) and an array 1 (a volume set) on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
0	Mirror	15.0MB		None	1:02:0 64.0KB: 15.0MB
Tigris					1:03:0 64.0KB: 15.0MB
E: 1	Volume	15.0MB		NTFS	1:00:0 64.0KB: 15.0MB

The following example reconfigures a volume set to a stripe set:

```
AAC0>container reconfigure /partition_size=1 /stripe 1
```

```
Executing: container reconfigure /partition_size=1 /stripe=TRUE 1
```

As the command executes, note that the title bar of the DOS window displays the status of the command. For example:

```
Stat:OK!Task:100,Func:RCF Ctr:1,State:RUN 97.2%
```

See [page 3-10](#) for detailed information on the status information that the command displays in the title bar as it executes the task.



Use the `container list` command after using the `container reconfigure` command to display information about the array you just reconfigured, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type Size Ctr Size Usage C:ID:L Offset:Size					
0	Mirror	15.0MB		None	1:02:0 64.0KB: 15.0MB
Tigris					1:03:0 64.0KB: 15.0MB
E: 1	Stripe	15.0MB		NTFS	1:00:0 64.0KB: 15.0MB

As the result of reconfiguring an array (in this example, array 1) with the `container reconfigure` command, the **Type** column displays **Stripe** (instead of **Volume**) to indicate that the specified array is now a stripe set instead of a volume set.

## Related Information

container commands:

- `container create mirror` ([page 5-5](#))
- `container create raid5` ([page 5-24](#))
- `container create stripe` ([page 5-30](#))
- `container create volume` ([page 5-35](#))
- `container extend file_system` ([page 5-43](#))
- `container list` ([page 5-54](#))

## container release\_cache

To release the cache buffers associated with a specific array, use the `container release_cache` command. When a disk fails or you remove a disk, the cache buffers associated with specific arrays remain locked on the controller. This allows the opportunity for the disk to come back online or gives you the opportunity to replace the disk. In this case the arrays regain the previously locked buffers.

On the other hand, you may want to return these cache buffers to the global buffer pool. This command allows you to disassociate the cache buffers associated with specific arrays and release these cache buffers to the global pool.

### Syntax

```
container release_cache {container}
```

### Parameters

{container}

Specifies the ID number (0 to 63) of the array on which you want to release cache buffers.

### Examples

The following example releases the cache buffers associated with array 0:

```
AAC0>container release_cache 0  
Executing: container release_cache 0
```

### Related Information

container commands:

- `container set cache` ([page 5-97](#))

## container remove drive\_letter

To remove a drive letter, use the `container remove drive_letter` command. To use the `container remove drive_letter` command, none of the array's files can be open.

### Command Availability

This command and its associated switches (if any) are supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)

### Syntax

```
container remove drive_letter {string}
```

### Parameters

```
{string}
```

Specifies the drive letter to remove. The colon (:) after the drive letter is optional.

### Examples

Before removing a drive letter from an array, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a volume set) on this controller with a drive letter F:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume	10.0MB		NTFS	0:02:0 64.0KB: 10.0MB

The following example removes drive letter F from array 0:

```
AAC0> container remove drive_letter F
```

```
Executing: container remove drive_letter "F"
```

Use the `container list` command after using the `container remove drive_letter` command to display information about the array on which you just removed a drive letter, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition			
Dr	Label	Type	Size	Ctr	Size	Usage	C:ID:L	Offset:Size
0	Volume	10.0MB		NTFS	0:02:0	64.0KB:	10.0MB	

As the result of removing a drive letter from an array (in this example, array 0) with the `container remove drive_letter` command:

- The **Dr** column no longer displays the drive letter assigned in a previous use of the `container assign drive_letter` command. In this example, the letter **F** no longer appears in the **Dr** column.

## Related Information

container commands:

- `container list` ([page 5-54](#))

## container remove failover

To remove one or more failover disks that were assigned to an array using the `container set failover` command, use the `container remove failover` command.

### Syntax

```
container remove failover {container}  
{scsi_device} [{scsi_device}...]
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array whose assigned failover disk(s) you want to remove.

`{scsi_device}`

Specifies the ID for the SCSI device that you want to remove as a failover disk for the array specified in the `container` parameter. You previously assigned this SCSI device as a failover disk with the `container set failover` command. See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

`{scsi_device}...`

Specifies the ID(s) for any other SCSI device(s) you want to remove as failover disk(s) for the array specified in the `container` parameter.

## Examples

Before removing one or more failover disks from an array, use the `container show failover` command to obtain information about any existing failover disks assigned to arrays.

As the following example shows, there is an array 0 that has two failover disks assigned to it on this controller:

```
AAC0>container show failover

Executing: container show failover
container Scsi C:ID:L
-----
0          0:03:0  0:04:0
```

The following example removes SCSI device 0:03:0 as an assigned failover disk for array 0:

```
AAC0>container remove failover 0 (0,3,0)

Executing: container remove failover 0 (CHANNEL=0,ID=3,LUN=0)
```

Use the `container show failover` command after removing a failover disk from an array's failover disk list with the `container remove failover` command to display information about the array's failover disk list, as in the following example:

```
AAC0>container show failover

Executing: container show failover
container Scsi C:ID:L
-----
0          0:04:0
```

As the display shows, SCSI device (0,03,0) is no longer in the failover disk list for array 0.

## Related Information

container commands:

- `container set failover` ([page 5-100](#))
- `container show failover` ([page 5-112](#))

## container remove file\_system

To remove the file system from an array, use the `container remove file_system` command. Use this command when the file system on the array is no longer needed.

This command fails if there are open files on the array.

### Command Availability

This command and its associated switches (if any) are supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)

### Syntax

```
container remove file_system {container}
```

### Parameters

{container}

Specifies the ID number (0 to 63) of the array from which you want to remove the file system.

### Examples

Before removing a file system from an array, use the `container list` command to obtain information about any existing arrays.

As the following example shows, there is an array 0 (a volume set) on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume	10.0MB		NTFS	0:02:0 64.0KB: 10.0MB

The following example removes the file system from array 0:

```
AAC0>container remove file_system 0
```

```
Executing: container remove file_system 0
```

Use the `container list` command after removing a file system from an array with the `container remove file_system` command to display information about the container, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume 10.0MB			None	0:02:0 64.0KB: 10.0MB

As a result of removing a file system from an array, the **Usage** column displays **None** (instead of NTFS) to indicate that the specified array no longer has a file system.

## Related Information

container commands:

- `container format` ([page 5-54](#))
- `container list` ([page 5-54](#))



## container remove global\_failover

To remove one or more failover disks that were assigned to all arrays through the `container set global_failover` command, use the `container remove global_failover` command.

### Syntax

```
container remove global_failover {scsi_device}
[ {scsi_device} ... ]
```

### Parameters

```
{scsi_device}
```

Specifies the ID for the SCSI device that you want to remove as a failover disk for all arrays. You previously assigned this SCSI device as a failover disk with the `container set global_failover` command. See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

```
{scsi_device}...
```

Specifies the ID(s) for any other SCSI device(s) you want to remove as failover disk(s) for all arrays. You previously assigned these SCSI device(s) as failover disk(s) with the `container set global_failover` command.

### Examples

Before removing one or more failover disks from all arrays, use the `container show failover` command to obtain information about any existing failover disks assigned to arrays.

As the following example shows, there is an array 0 that has devices (1,2,0) and (1,3,0) assigned to it.

```
AAC0>container show failover
```

```
Executing: container show failover
container Scsi C:ID:L
```

```
-----
GLOBAL  1:02:0  1:03:0
0        --- No Devices Assigned ---
```

The following example removes SCSI device 1:02:0 as an assigned failover disk for all arrays:

## *container Commands*

```
AAC0>container remove global_failover (1,2,0)
```

```
Executing: container remove global_failover (CHANNEL=1,ID=2,LUN=0)
```

Use the `container show failover` command after removing a failover disk from an array's failover disk list with the `container remove failover` command to display information about the array's failover disk list, as in the following example:

```
AAC0>container show failover
```

```
Executing: container show failover
```

```
container Scsi C:ID:L
```

```
-----  
GLOBAL  1:03:0  
0        --- No Devices Assigned ---
```

As the display shows, SCSI device (1,2,0) is no longer in the failover disk list for all arrays. However, SCSI device (1,3,0) remains in the failover disk list for all arrays.

## **Related Information**

container commands:

- `container set failover` ([page 5-100](#))
- `container set global_failover` ([page 5-102](#))
- `container show failover` ([page 5-112](#))

## container restore RAID5

To restore a RAID-5 set, use the `container restore RAID5` command. Typically, you use this command to restore a RAID-5 set that contains one or more dead partitions. This command does not always succeed and, in fact, you should use this command if all other measures to restore the RAID-5 set fail.

One situation where the command might prove useful is for those situations where you may have inadvertently pulled a drive and then almost immediately put it back. The act of pulling the drive may cause some dead partitions to appear. You can then restore the RAID-5 set by using this command.

### Syntax

```
container restore RAID5 {container}
```

### Parameters

{container}

Specifies the ID number (0 to 63) of the array (a RAID-5 set) to restore.

### Examples

Before restoring a RAID-5 set, use the `container list` command to display information about the RAID-5 set. If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column changes to a “!” (exclamation point).

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe		Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L	Offset:Size
0	RAID-5	30.0MB	64KB	None	0:02:0	64.0KB! 10.0MB
					0:03:0	64.0KB: 10.0MB
					0:04:0	64.0KB: 10.0MB
					0:05:0	64.0KB: 10.0MB

The previous example shows that there is one dead partition on this RAID-5 set. The following example uses the `container restore RAID5` command to attempt to restore the RAID-5 set:

```
AAC0>container restore RAID5 0
```

```
Executing: container restore RAID5 0
```

Use the `container list` command after using the `container restore RAID5` command to display information about the RAID-5 set:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type Size Ctr Size Usage C:ID:L Offset:Size					
0	RAID-5	30.0MB	64.0KB	None	0:02:0 64.0KB: 10.0MB 0:03:0 64.0KB: 10.0MB 0:04:0 64.0KB: 10.0MB 0:05:0 64.0KB: 10.0MB

Note that in this case the command successfully restored the RAID-5 set as indicated by the replacement of the exclamation point ("!") with a colon (":").

## Related Information

container commands:

- `container create raid5` ([page 5-24](#))
- `container list` ([page 5-54](#))

disk commands:

- `disk remove dead_partitions` ([page 7-9](#))

## container scrub

To scrub a redundant array, use the `container scrub` command. A mirror set, a multilevel array of mirror sets, and a RAID-5 set are examples of redundant arrays. For a mirror set or a multilevel array of mirror sets, this means the command reconstructs the data on both mirror halves (partitions), if found to be different. For RAID-5 sets, the command recalculates and replaces, if necessary, the parity information.

### Syntax

```
container scrub [/io_delay{=integer}] [/no_repair{=boolean}] [/wait{=boolean}] {container}
```

### Parameters

{container}

Specifies the ID number (0 to 63) of the redundant array to scrub.

### Switches

/io\_delay{=integer}

Specifies the number of milliseconds the controller waits between the I/O operations required to scrub the redundant array. The I/O delay value is not preserved between reboots of the operating system. Valid values are 0 through 100.

/no\_repair{=boolean}

Specifies whether the command performs the scrub operation on the redundant array without repairing the error. If you set this switch to `TRUE`, the command performs the scrub action without repairing the error. If you set this switch to `FALSE`, the command performs the scrub action and repairs any errors.

/wait{=boolean}

Specifies whether the command performs the scrub action synchronously or asynchronously. If you set this switch to `TRUE`, the command performs the scrub action synchronously, which means the command prompt does not return until the scrub action completes. If you set this switch to `FALSE`, the command performs the scrub action asynchronously, which means the command prompt returns immediately.

## Examples

Before scrubbing a redundant array, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a mirror set) on this controller:

```
AAC0>container list

Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
D: 0	Mirror	10.0MB		NTFS	0:02:0 64.0KB: 10.0MB
					0:02:0 10.0MB: 10.0MB

The following example synchronously scrubs array 0 using a delay time of 5 milliseconds:

```
AAC0>container scrub /io_delay=5 /wait=TRUE 0

Executing: container scrub /io_delay=5 /wait=TRUE 0
```

As the command executes, note the title bar of the DOS window displays the status of the command. For example:

```
Stat:OK!Task:100,Func:MSC Ctr:0,State:RUN 97.2%
```

See [page 3-10](#) for detailed information on the status information that the command displays in the title bar as it executes the task.

Use the `container list` command after using the `container scrub` command to display information about the scrubbed array, as in the following example:

```
AAC0>container list

Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
D: 0	Mirror 10.0MB			NTFS	0:02:0 64.0KB: 10.0MB
					0:02:0 10.0MB: 10.0MB

As the example shows, there is no difference between the array list display prior to and after the scrub operation.

## Related Information

container commands:

- `container create mirror` ([page 5-5](#))
- `container create mmirror` ([page 5-9](#))
- `container create raid5` ([page 5-24](#))
- `container list` ([page 5-54](#))

## container set cache

To set cache parameters for a specific array, use the `container set cache` command. You can use this command only if a native operating system's file system—for Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only), NTFS, or FAT—resides on the array. In fact, the CLI prevents you from setting cache parameters if the array:

- Is an array involved in a reconfiguration operation

The CLI disables the cache parameters on the array involved in a reconfiguration operation. The CLI re-enables the cache parameters (assuming they were previously set) on the array when the reconfiguration operation completes.

**The controller provides two global cache buffer pools available to arrays: a volatile read-ahead cache and a nonvolatile NVRAM write-back cache. Collectively, these global caches are referred to as the array cache.**

The `container set cache` command allows you to set several characteristics associated with the array cache. These characteristics are embodied in the switches for the command.

### Notes

**Some controllers may not support the NVRAM write-back cache. Additionally, some controllers may not support the ability to enable the NVRAM write-back cache.**

### Syntax

```
container set cache [/read_cache_enable{=boolean}]
[/write_cache_enable{=boolean}] {container}
```



## Parameters

{container}

Specifies the ID number (0 to 63) of the array on which you want to set cache parameters.

## Switches

/read\_cache\_enable{=boolean}

Specifies whether to enable the read-ahead cache. If you set this switch to **TRUE**, the command enables the read-ahead cache for the specified array. This switch should always be enabled to optimize performance, unless your application--which is unlikely--is doing completely random reads.

This switch defaults to **TRUE**.

To disable the read-ahead cache for the specified array, set this switch to **FALSE**. Note that if you disable the read-ahead cache, no other characteristics can be set.

/write\_cache\_enable{=boolean}

Specifies whether to enable the array's NVRAM write-back cache.

## Examples

Before setting cache parameters for a specific array, use the `container list` command to obtain information about any existing arrays.

As the following example shows, there is an array 0 (a volume set) on this controller with no file system on it (as evidenced by **None** displayed in the **Usage** column):

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe		Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L	Offset:Size
0	Volume	15.0MB		None	0:02:0	64.0KB: 15.0MB

The following example sets cache parameters for array 0 by accepting all of the defaults:

```
AAC0>container set cache 0
```

Executing: container set cache 0

After setting cache parameters for a specific array, use the `container show cache` command to display cache parameter information, as in the following example:

AAC0>container show cache 0

Executing: container show cache 0

Global container Read Cache Size : 5345280

Global container Write Cache Size : 1970176

Read Cache Status : ENABLED

Write Cache Status : ENABLED

Stream Detection Status : ENABLED

## Related Information

container commands:

- `container release_cache` ([page 5-83](#))
- `container show cache` ([page 5-109](#))

## container set failover

To assign an automatic failover disk(s) for a single array, use the `container set failover` command. If the array was already assigned a failover disk(s), the command adds the specified disk(s) to the array's list of failover disk(s). Although all array types accept failover assignments, only mirror set and RAID-5 set (redundant) array types use the failover assignment if a disk fails.

### Syntax

```
container set failover {container} {scsi_device}
[ {scsi_device} ... ]
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array on which to assign an automatic failover disk(s).

`{scsi_device}`

Specifies the ID for the SCSI device that you want to assign as a failover disk to the array specified in the `container` parameter. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), a SCSI device ID (0 through 15 inclusive), and a SCSI device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

`{scsi_device} ...`

Specifies the ID(s) for additional SCSI device(s) that you want to assign as failover disk(s) to the array specified in the `container` parameter.

### Examples

Before assigning a SCSI device as a failover disk to an array, use the `container list` command to obtain information about any existing arrays.

As the following example shows, there is an array 0 (a mirror set) on this controller:

```
AAC0>container list
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
D: 0	Mirror	10.0MB		NTFS	0:02:0 64.0KB: 10.0MB
					0:02:0 10.0MB: 10.0MB

The following example assigns two SCSI failover disks to array 0. If you assign only one SCSI disk as a failover device, ensure that there is sufficient freespace. If you assign more than one SCSI disk as failover devices, you need ensure that only one of these disks has sufficient space. The reason for this is that the failover operation will look for the disk that has enough space.

```
AAC0>container set failover 0 (0,3,0) (0,4,0)
Executing: container set failover 0 (CHANNEL=0,ID=3,LUN=0)
(CHANNEL=0,ID=4,LUN=0)
```

Use the `container show failover` command after using the `container set failover` command to display information about the array just assigned failover disk(s).

## Related Information

container commands:

- `container list` ([page 5-54](#))
- `container remove failover` ([page 5-86](#))
- `container show failover` ([page 5-112](#))

## container set global\_failover

To assign an automatic failover disk(s) for all arrays, use the `container set global_failover` command. If the array was already assigned a failover disk(s), the command adds the specified disk(s) to the array's list of failover disk(s). Although all array types accept failover assignments, only mirror set and RAID-5 set (redundant) array types use the failover assignment if a disk fails.

### Syntax

```
container set global_failover {scsi_device}
[ {scsi_device} ... ]
```

### Parameters

```
{scsi_device}
```

Specifies the ID for the SCSI device that you want to assign as a failover disk. The command assigns this disk to all arrays. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), a SCSI device ID (0 through 15 inclusive), and a SCSI device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

```
{scsi_device} ...
```

Specifies the ID(s) for additional SCSI device(s) that you want to assign as failover disk(s) to all arrays.

### Examples

Before assigning a SCSI device as a failover disk to all arrays, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a mirror set) on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe		Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L	Offset:Size
0	Mirror	15.0MB		None	1:02:0	64.0KB: 15.0MB
Tigris					1:03:0	64.0KB: 15.0MB

The following example assigns two SCSI failover disks to all arrays. If you assign only one SCSI disk as a failover device, ensure that there is sufficient freespace. If you assign more than one SCSI disk as failover devices, you need ensure that only one of these disks has sufficient space. The reason for this is that the failover operation will look for the disk that has enough space.

```
AAC0>container set global_failover (1,2,0) (1,3,0)
```

```
Executing: container set global_failover (CHANNEL=1,ID=2,LUN=0)
(CHANNEL=1,ID=3,LUN=0)
```

Use the `container show failover` command (with or without the `/global switch`) after using the `container set failover` command to display information about the array(s) just assigned failover disk(s), as in the following example:

```
AAC0>container show failover /global
```

```
Executing: container show failover /global=TRUE
container Scsi C:ID:L
```

```
-----
GLOBAL  1:02:0  1:03:0
```

As the example shows, the display indicates that devices (1,2,0) and (1,3,0) are assigned as failover devices for all arrays (the word **GLOBAL** appears in the **array** column).

## Related Information

container commands:

- `container list` ([page 5-54](#))
- `container remove failover` ([page 5-86](#))
- `container remove global_failover` ([page 5-90](#))
- `container set failover` ([page 5-100](#))
- `container show failover` ([page 5-112](#))

## container set io\_delay

To set the I/O delay for an array, use the `container set io_delay` command. You can specify an I/O delay for the following array-related tasks:

- Creating a mirror set
- Creating a multilevel array of mirror sets from a multilevel array of volume sets
- Scrubbing a redundant array

You use this command to change the I/O delay from the one you specified in the command for the previously listed tasks. The I/O delay value is not preserved between reboots of the operating system.

### Syntax

```
container set io_delay {container} {integer}
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array on which to set the I/O delay.

`{integer}`

Specifies the number of milliseconds the controller waits between the I/O operations required to perform the specified background task. The I/O delay value is not preserved between reboots of the operating system. Valid values are 0 through 100.

## Examples

Before setting the I/O delay on an array, use the `container list` command to obtain information about any existing arrays.

As the following example shows, there is an array 0 (a volume set) on this controller:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume	10.0MB		NTFS	0:02:0 64.0KB: 10.0MB

The following example creates a mirror set asynchronously and sets an I/O delay of 20 milliseconds for array 0. You need to perform the create mirror task asynchronously. Otherwise, you cannot change the I/O delay:

```
container create mirror /io_delay=20 0 (0,2,0)
```

```
Executing container create mirror /io_delay=20 0 (CHANNEL=0,ID=2,LUN=0)
```

The following example shows how to change the I/O delay to 40 milliseconds on the currently running create mirror task:

```
AAC0>container set io_delay 0 40
```

```
Executing: container set io_delay 0 40
```

Use the `container list` command after using the `container set io_delay` command to display information about the array, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Volume	10.0MB		NTFS	0:02:0 64.0KB: 10.0MB

As the example shows, there is no difference between the array list display prior to and after the set I/O delay operation.



## **Related Information**

container commands:

- container create mirror ([page 5-5](#))
- container create mmirror ([page 5-9](#))
- container list ([page 5-54](#))
- container scrub ([page 5-94](#))

## container set label

To assign a new label to the specified array or to assign a label to an array that has no label, use the `container set label` command.

The following commands have a `/label` switch that allows you to assign a label to the array when you create it:

- `container create mstripe` (a multilevel stripe set)
- `container create mvolume` (a multilevel volume set)
- `container create raid5` (a RAID-5 set)
- `container create stripe` (a stripe set)
- `container create volume` (a volume set)

The label you specify with the `container set label` command and the `/label` switch associated with the previously listed `container create` commands is not the label that displays in Windows Explorer. The label displayed by Windows Explorer comes from the label specified with the `container format` command.

### Syntax

```
container set label {container} {string}
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array to which you want to assign a label.

`{string}`

Specifies the label you want to assign to the array. The label is a string of up to sixteen characters.

### Examples

Before assigning a label to a specific array, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is an array 0 (a volume set) on this controller with no label assigned to it (as evidenced by no label in the **Num Label** column):

```
AAC0>container list
```

```
Executing: container list
  Num      Total  Oth Stripe      Scsi  Partition
Dr Label Type  Size  Ctr Size  Usage  C:ID:L Offset:Size
-----
  0      Volume 15.0MB      None  0:02:0 64.0KB: 15.0MB
```

The following example assigns the label **Tigris** to array 0:

```
AAC0>container set label 0 Tigris
```

```
Executing: container set label 0 "Tigris"
```

Use the **container list** command after using the **container set label** command to display the new label, as in the following example:

```
AAC0>container list
Executing: container list
  Num      Total  Oth Stripe      Scsi  Partition
Dr Label Type  Size  Ctr Size  Usage  C:ID:L Offset:Size
-----
  0      Volume 15.0MB      None  0:02:0 64.0KB: 15.0MB
  Tigris
```

As the example shows, the new label **Tigris** now appears in the **Num Label** column.

## Related Information

container commands:

- **container create mstripe** ([page 5-14](#))
- **container create mvolume** ([page 5-19](#))
- **container create raid5** ([page 5-24](#))
- **container create stripe** ([page 5-30](#))
- **container create volume** ([page 5-35](#))
- **container format** ([page 5-54](#))

## container show cache

To display the current cache parameters associated with a specific array, use the `container show cache` command. Typically, you use this command after setting cache parameters for a specific array using the `container set cache` command.

### Syntax

```
container show cache {container}
```

### Parameters

```
{container}
```

Specifies the ID number (0 to 63) of the array whose associated cache parameters you want to display.

### Examples

Before setting cache parameters for a specific array, use the `container list` command to obtain information about any existing arrays.

As the following example shows, there is an array 0 (a volume set) on this controller with no file system on it (as evidenced by **None** displayed in the **Usage** column). Assume the cache parameters for this array were previously set with the `container set cache` command:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
0	Volume	15.0MB		None	0:02:0 64.0KB: 15.0MB

The following example displays cache parameters for array 0:

```
AAC0>container show cache 0

Executing: container show cache 0

Global container Read Cache Size : 5345280
Global container Write Cache Size : 1970176

Read Cache Setting      : ENABLED
Write Cache Setting     : ENABLED
Write Cache Status      : ENABLED
```

The following list provides a brief description of the items that appear as a result of using the `container show cache` command:

- **Global container Read Cache Size**  
This item indicates the number of blocks devoted to the read cache.
- **Global container Write Cache Size**  
This item indicates the number of blocks devoted to the write cache.
- **Read Cache Setting**  
This item indicates whether the read cache is set for the array. The value `ENABLED` indicates the read cache is set for this array. The value `DISABLED` indicates the read cache is disabled for this array.
- **Write Cache Setting**  
This item indicates whether the write cache is set for the array. The value `ENABLED` indicates the write cache is set for this array. The value `DISABLED` indicates the write cache is disabled for this array.
- **Write Cache Status**  
This item indicates the current status of the write cache for the array.

## Related Information

`container` commands:

## *container Commands*

- `container list` ([page 5-54](#))
- `container set cache` ([page 5-97](#))
- `container show cache` ([page 5-109](#))

## container show failover

To display a list of failover disks assigned to an array, use the `container show failover` command. You assign failover disks to an array with the `container set failover` command.

### Syntax

```
container show failover [{container}]
```

### Parameters

`{container}`

Specifies the ID number (0 to 63) of the array whose assigned failover disk(s) you want to display. If you do not specify this parameter, the command displays all arrays and their assigned failover disk(s).

### Examples

The following example shows the failover disks assigned to array 0:

```
AAC0>container show failover 0

Executing: container show failover 0
container Scsi C:ID:L
-----
0          0:03:0  0:04:0
```

The example shows that array 0 was assigned disks 0:03:0 and 0:04:0 as failover disks.

### Related Information

container commands:

- `container remove failover` ([page 5-86](#))
- `container set failover` ([page 5-100](#))

## container split

To split a mirror set or a multilevel array of mirror sets into two separate single-partition volume sets or two multilevel arrays of single-partition volume sets, use the `container split` command. Once this command completes execution, the two volume sets cannot be merged.

Before using this command to split an array, shut down system applications (such as, databases) in order to flush application data to the controller.

When you split a mirror set or a multilevel array of mirror sets, the system creates a new (referred to as the split) array. The split array contains half of the storage data from the original (referred to as the master) array (mirror set). In addition, the split array has an identical copy of the data from the old disk at the time of the split operation.

On Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only), the split array's file system may be inconsistent and you should verify and repair it as follows:

- Assign a drive letter to the split array using the `container assign drive_letter` command.
- Run the Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only) `CHKDSK` command with the `/F` switch. See your Windows documentation for information on how to use the `CHKDSK` command.

On UnixWare, before splitting a mirror set that contains a volume table of contents, run the `edvtoc` command and change the volume label to a series of twelve spaces. You can then use the `container split` command to split the mirror set.

### Syntax

```
container split {container}
```



## Parameters

{container}

Specifies the ID number (0 to 63) of the array (mirror set or multilevel array of mirror sets) to split into two separate arrays.

## Examples

Before splitting a mirror set or a multilevel array of mirror sets, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is one existing array (array 0, a mirror set) on this controller at the time the mirror set is split:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
-----					
F: 0	Mirror	10.0MB		NTFS	0:02:0 64.0KB: 10.0MB
					0:02:0 10.0MB: 10.0MB

The following example shows how to split a mirror set:

```
AAC0>container split 0
```

```
Executing: container split 0
```

```
container 1 created
```

On UNIX systems, the message displayed after you execute the `container split` command includes the root special file associated with the newly created split array.

Use the `container list` command after using the `container split` command to display information about the split mirror set, as in the following example:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	Partition
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Offset:Size
F: 0	Volume 10.0MB			NTFS	0:02:0 64.0KB: 10.0MB
1	Volume 10.0MB			NTFS	0:02:0 10.0MB: 10.0MB

The following list describes the change to the display as the result of splitting a mirror set with the `container split` command:

- The **Num Label** column displays the IDs (in the example, 0 and 1) of the split mirror set. array 0 is the master array; array 1 is the split array.

This column also displays the label(s) assigned to the array(s) when the array(s) was created. If no label(s) was assigned to the array(s) then no label(s) appears in the column.

On UNIX systems, the root special file associated with the array also appears in this column.

- The **Type** column displays **Volume** as the array type for a split mirror set.
- The **Usage** column displays NTFS to indicate that file systems exist on both arrays. You can create an NTFS or FAT file system on an array by using the `container format` command.
- The **Scsi C:ID:L** column displays the SCSI device IDs for the disk on which the split mirror set (now two volume sets) reside. In the example, the SCSI device ID is 0:02:0.
- The **Partition Offset:Size** column displays the partition offsets and sizes for the partition associated with the split mirror set (now two volume sets).

If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column changes to a “!” (exclamation point). See the `disk remove dead_partitions` (page 7-9) command for more information on dead partitions.

## **Related Information**

container commands:

- container create mirror ([page 5-5](#))
- container create mmirror ([page 5-9](#))
- container list ([page 5-54](#))
- container unmirror ([page 5-119](#))

disk commands:

- disk remove dead\_partitions ([page 7-9](#))

## container unlock



**Caution:** Use the `container unlock` command only under the direction of Technical Support.

To unlock an array so it can be moved, deleted, made read-only, and used to create a multilevel array, use the `container unlock` command.

### Syntax

```
container unlock {container}
```

### Parameters

```
{container}
```

Specifies the ID number (0 to 63) of the array to unlock.

### Examples

Before unlocking an array, use the `container list` command (with the `/full` switch) to obtain information about any existing arrays. As the following example shows, there is an array 0 (a volume set) on this controller. Note that the **Lk** column displays an **L**, which indicates that array 0 was previously locked with the `container lock` command:

```
AAC0>container list /full
Executing: container list /full=TRUE
  Num      Total  Oth Stripe      Scsi
Dr Label Type  Size   Ctr Size   Usage  C:ID:L  Lk
--  ----  -
F: 0      Volume 10.0MB                NTFS   0:02:0  L
```

Note that the example eliminates some items in the `container list` display so that you can see an example of the **Lk** column.

The following example shows how to unlock array 0:

```
AAC0>container unlock 0
```

```
Executing: container unlock 0
```

Use the `container list` command (with the `/full` switch) after using the `container unlock` command to display information about the unlocked array, as in the following example:

```
AAC0>container list /full
```

```
Executing: container list /full=TRUE
```

Num	Total	Oth	Stripe	Scsi	
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L Lk
-----					
F: 0	Volume 10.0MB			NTFS	0:02:0

Note that the example eliminates some items in the `container list` display so that you can see an example of the **Lk** column.

As a result of unlocking an array (in this example, array 0) with the `container unlock` command, the **Lk** column displays a blank (instead of an L) to indicate that the specified array is now unlocked.

## Related Information

container commands:

- `container list` ([page 5-54](#))
- `container lock` ([page 5-65](#))

## container unmirror

To unmirror a mirror set, resulting in a single-partition volume set and freespace, use the `container unmirror` command. Note that there is no equivalent command for a multilevel array of mirror sets. To unmirror a multilevel array of mirror sets, use this command for each underlying mirror set.

### Syntax

```
container unmirror {container}
```

### Parameters

```
{container}
```

Specifies the ID number (0 to 63) of the mirror set to unmirror. You previously created the mirror set by using the `container create mirror` or `container create mmirror` command.

### Examples

Before unmirroring a mirror set, use the `container list` command to obtain information about any existing arrays. As the following example shows, there is one existing array (array 0, a mirror set) on this controller at the time the mirror set is unmirrored:

```
AAC0>container list
```

```
Executing: container list
```

Num	Total	Oth	Stripe	Scsi	
Dr Label Type	Size	Ctr	Size	Usage	C:ID:L
0	Mirror	10.0MB		NTFS	0:02:0 0:03:0

Note that the example eliminates some items in the `container list` display so that you can see an example of the **State** column.

The following example shows how to unmirror a mirror set from the mirror set that resides on SCSI device (0,3,0):

```
AAC0>container unmirror 0
```

```
Executing: container unmirror 0
```

Note that the example eliminates some items in the `container list` display so that you can see an example of the **State** column.

The following list describes the changes to the display as a result of using the `container unmirror` command:

- The **Type** column displays **Volume**, which indicates that the mirror set was created from a Volume set.
- The **State** column displays a blank space instead of Normal. The Normal state is displayed only for mirror sets.
- The **Scsi C:ID:L** column displays only one SCSI ID.
- The **Partition Offset:Size** column displays only one partition offset and size.

If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column changes to a “!” (exclamation point). See the `disk remove dead_partitions` ([page 7-9](#)) command for more information on dead partitions.

## Related Information

`container` commands:

- `container create mirror` ([page 5-5](#))
- `container list` ([page 5-54](#))

`disk` commands:

- `disk remove dead_partitions` ([page 7-9](#))

# 6

## controller Commands

### In this Chapter

<i>controller details</i>	6-2
<i>controller firmware compare</i>	6-6
<i>controller firmware save</i>	6-7
<i>controller firmware update</i>	6-9
<i>controller list</i>	6-10
<i>controller pause_io</i>	6-12
<i>controller rescan</i>	6-15
<i>controller reset_scsi_channel</i>	6-16
<i>controller resume_io</i>	6-17
<i>controller set automatic_failover</i>	6-18
<i>controller set array_verify</i>	6-20
<i>controller show automatic_failover</i>	6-22
<i>controller show channels</i>	6-23
<i>controller show array_verify</i>	6-25



## controller details

To display details about the currently opened controller and software, use the `controller details` command. These details include the controller type and software revision levels.

### Syntax

```
controller details
```

### Examples

The following example shows sample output as a result of using the `controller details` command:

```
AAC0> controller details

Executing: controller details
Controller Information
-----
      Remote Computer: .
      Device Name: AAC0
      Controller type: Adaptec 2410SA
      Access Mode: READ-WRITE
Controller serial number: Last six digits = 8A277A
      Number of Buses: 1
      Devices per Bus: 15
      Controller CPU: i960
      Controller CPU speed: 100 Mhz
      Controller Memory: 144 Mbytes
      Battery State: Not present
Component Revisions
-----
      CLI: 3.0-0 (Build #5523)
      API: 3.0-0 (Build #5523)
      Miniport Driver: 3.0-0 (Build #5131)
Controller Software: 3.0-0 (Build #5478)
      Controller BIOS: 3.0-1 (Build #5523)
Controller Firmware: (Build #5478)
Controller Hardware: 3.3
```

The following sections provide more information on the items that the `controller details` command displays.

#### Device Name Item

The `Device Name:` item displays the name of the controller. This is the controller name specified when you opened the controller with the `open` command. In the example, the controller name is AAC0.

### **The Controller type Item**

The `Controller type`: item displays the controller type. In the example, the controller type is Adaptec 5400S.

### **The Access Mode Item**

The `Access Mode`: item displays the access mode of the controller. The command displays the value `READ-WRITE` if you opened the controller for read-write operations. Otherwise, it displays `READ-ONLY` if you opened the controller for read-only operations.

### **The Controller serial number Item**

The `Controller serial number`: item displays the controller serial number. The command displays the last six hexadecimal characters of the controller serial number. These six hexadecimal characters match the last six characters displayed on the “SN =” sticker located on the controller itself.

### **The Number of Channels and Devices per Channel Items**

The `Number of Buses`: item displays the number of channels actually found on the controller. In the example, the number of channels on the controller is four.

The `Devices per Bus`: item displays the number of SCSI disk devices that can be put onto each SCSI channel for this controller. In the example, 15 SCSI disks can be put onto each SCSI channel.

### **The Controller CPU Item**

The `Controller CPU`: item displays the type of central processing unit (CPU) that the controller uses.

### **The Controller CPU Speed and Controller Memory Items**

The `Controller CPU Speed`: item (if available) displays the clock speed (in megahertz) of the CPU that resides on the controller.

The `Controller Memory`: item displays the total amount of memory on the controller that programs and buffer data can use. In the example, the total memory is 48 MB.

### The Battery State Item

The `Battery State:` shows that this controller is not equipped with a battery.

### The Component Revisions Items

Table 6-1 describes each of the `Component Revisions:` items. This information is useful if you ever need to contact a technical support representative.

Table 6-1. Component Revision Items

Component Revision Item	Meaning
CLI:	Displays the revision level of the Command Line Interface. In the example, the revision level is 3.0-0 (Build #5523).
API:	Displays the revision level of the controller's application programming interface (API). In the example, the revision level is 3.0-0 (Build #5478).
Service:	Displays the revision level of the network services software. This item displays only if the open controller resides on a remote computer. In the example, the revision level is 2.1-0 (Build #2881).
Remote API:	Displays the revision level of the API that the network services software uses on the remote computer. This item displays only if the open controller resides on a remote computer. In the example, the revision level is 2.1-0 (Build #2881).
Miniport Driver:	Displays the version number of the miniport portion of the AAC.SYS device driver. This item displays only if the controller runs the AAC.SYS device driver. In the example, the revision level is 3.0-0 (Build #5131).
Controller Software:	Displays the revision level of the controller software. In the example, the revision level is 3.0-0 (Build #5478).
Controller BIOS:	Displays the revision level of the controller BIOS. In the example, the revision level is 3.0-1 (Build #5523).

Table 6-1. Component Revision Items (Continued)

Component Revision Item	Meaning
Controller Firmware:	Displays the revision level of the firmware. In the example, the revision level is (Build #5478).
Controller Hardware:	Displays the version of the controller hardware. In the example, the version is 3.3.

## Related Information

General control commands:

- open ([page 4-3](#))

## controller firmware compare

To compare the contents of each of the flash components on a controller to the corresponding image in a pair of user flash image (UFI) files and indicate whether they match, use the `controller firmware compare` command.

### Syntax

```
controller firmware compare  
[/C[{controller_ID}][{controller_ID}...]  
[/D{UFI_file_path}]
```

### Switches

```
/C{controller_ID}
```

Specifies the controller ID representing the set of controllers on which to perform the firmware comparison.

If you do not specify this switch, this command compares the flash components on controller 0.

```
/D{UFI_file_path}
```

Specifies the path where the pair of UFI files are located.

If you do not specify this switch, this command compares the flash components against the pair of UFI files in the current default drive and directory.

### Examples

The following example compares the contents of a controller's flash components to the corresponding image in a pair of UFI files.

```
AAC0> controller firmware compare D:\
```

```
Executing: controller firmware compare "D:\"
```

```
The controller's firmware matches the firmware in the specified flash  
image files.
```

### Related Information:

controller commands:

- controller firmware save ([page 6-7](#))
- controller firmware update ([page 6-9](#))

## controller firmware save

To save the contents of a controller's flash in a pair of user flash image (EFI) files, use the `controller firmware save` command. The names of the pair of EFI files are based on the controller type and cannot be changed.

### Syntax

```
controller firmware save [/
C[{controller_ID}][{controller_ID}...] {/
D{EFI_file_path}]
```

### Switches

```
/C{controller_ID}
```

Specifies the controller ID representing the set of controllers on which to perform the firmware save.

If you do not specify this switch, this command saves the flash components on controller 0.

```
/D{EFI_file_path}
```

Specifies the path where the pair of EFI files are located. Use this switch to specify the drive and directory where you want to create the pair of EFI files.

If you do not specify this switch, this command creates the pair of EFI files in the current default drive and directory.

### Examples

The following example saves the contents of the opened controller's flash components to a pair of EFI files. In the example, one file is specified. A second file is also created with the same name, except that the number of file name suffix is incremented by 1.

```
AAC0> controller firmware save D:\aac\backup...
Executing: controller firmware save "D:\aac\backup..."
```

**Related Information:**

controller commands:

- controller firmware compare ([page 6-6](#))
- controller firmware update ([page 6-9](#))

## controller firmware update

To update a controller's flash components from the flash image data in a pair of user flash image (UFI) files, use the `controller firmware update` command. This command can update the flash components on a single controller or multiple controllers.

### Syntax

```
controller firmware update
[/C[{controller_ID}][{controller_ID}...]]
{/D{UFI_file_path}}
```

### Switches

```
/C{controller_ID}
```

Specifies the controller ID representing the set of controllers on which to perform the firmware update.

If you do not specify this switch, the firmware update is performed on controller 0.

```
/D{UFI_file_path}
```

Specifies the path where the pair of UFI files are located.

If you do not specify this switch, this command looks for or creates a pair of UFI files in the current default drive and directory.

### Examples

The following example updates a controller's firmware.

```
AAC0> controller firmware update D:\...
```

```
Executing: controller firmware compare "D:\..."
```

```
!!The controller's firmware has been successfully updated; you need to
restart the system!!.
```

### Related Information:

controller commands:

- controller firmware compare ([page 6-6](#))
- controller firmware save ([page 6-7](#))



## controller list

To list all controllers on a specified computer or display specific information about the currently opened controller, use the `controller list` command.

### Syntax

```
controller list [/domain{=string}] [{string}]
```

### Parameters

{string}

Specifies the computer name from which you want to display a list of controllers or specific information about the currently opened controller. If you do not specify a computer name, the command displays only the controllers on the local computer.

### Switches

/domain{=string}

Specifies the domain in which the specified computer resides.

If you do not specify this switch, the command assumes the local domain.

This switch is supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)

### Examples

The following example shows how to list all controllers on a local computer:

```
AAC0> controller list

Executing: controller list
Adapter Name      Adapter Type      Availability
-----
\\.\AAC0          Adaptec 5400S      read/write
```

The following sections provide descriptions of the items in the display.

### The Adapter Name and Adapter Type Items

The `Adapter Name` item displays the name(s) of all controllers on the local or remote computers. In the example, there is only one controller on the local computer and it is called AAC0.

The `Adapter Type` item displays the controller type. In the example, the controller type is Adaptec 5400S.

### The Availability Item

The `Availability` item displays the availability of the controller. Table 6-2 lists the values that can display in this item.

Table 6-2. Controller Availability Status

Controller Availability	Meaning
Unavailable	The controller is not available.
Unknown	The controller availability is unknown.
read only	The controller is available for read-only access.
read/write	The controller is available for read/write access. In the example, the controller is available for read/write access.

### Related Information

controller commands:

- controller details ([page 6-2](#))

## **controller pause\_io**

To pause all I/O activity on the currently opened controller, use the `controller pause_io` command. While I/O activity is paused, you can make changes to devices attached to the controller without rebooting. For example, you can add, remove, or change SCSI channel assignments.

Before using this command, you should consider the following points:

- The `controller pause_io` command allows a maximum delay of 150 seconds (that is, 2 1/2 minutes) for all I/O activity on the currently opened controller. This delay is based on an internal driver timeout (not on any network timeout) of three minutes.
- The network timeout is *client-based*, and different clients are likely to have different timeout values. Thus, pausing all I/O activity on the open controller can cause network timeouts, particularly if the specified delay is for more than one minute.
- If you find it necessary to pause the system for more than a minute, then network timeouts are usually preferable to rebooting.
- Taking the maximum delay of 2 1/2 minutes increases the chances that a network timeout will occur. If you cannot perform the hardware reconfiguration changes in less than 2 1/2 minutes, then you should shut down the system. Even if you take the maximum delay, you can usually resume the I/O sooner. This is the reason for making the pause I/O default time 2-1/2 minutes.
- The safest amount of time (that is, timeouts are unlikely to occur) to delay I/O operations is no more than 30 seconds.

## Command And Switch Availability

This command is supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)
- NetWare

## Notes

The controller automatically performs a rescan of the SCSI channel before the I/O is resumed.

Using this command when there is a pagefile on an array could cause unexpected behavior.

Use of the `controller pause_io` command is not permitted while an array task is running on the controller. (Use the `task list` command to display a list of currently running array tasks.)

## Syntax

```
controller pause_io [{integer}]
```

## Parameters

`{integer}`

Specifies the amount of time, in seconds, to cause the controller to wait before automatically resuming I/O.

The parameter defaults to the maximum value of 150 seconds. This is necessary because the `controller resume_io` command may not be able to access the controller under some circumstances.

## Examples

The following example pauses all I/O activity on the currently opened controller for 100 seconds:

```
AAC0> controller pause_io 100
```

```
Executing: controller pause_io 100
```

## **Related Information**

controller commands:

- controller resume\_io ([page 6-17](#))

task commands:

- task list ([page 9-2](#))

## **controller rescan**

To rescan the SCSI channels on the currently opened controller and update all underlying structures, use the `controller rescan` command. The effect of this command is to verify currently connected disks or to recognize new disks added to the channel.

This command rescans the disks for the arrays and loads the volatile array and the partition tables from disks again. If nothing on the disks has changed, `controller rescan` should have no visible effect. However, if any disk partition structures have changed, this command causes those changes to be reflected in the array and partition lists in the controller memory. If you add disks to the controller, the rescan enables you to use those disks.

### **Syntax**

```
controller rescan
```

## **controller reset\_scsi\_channel**

To reset a specific SCSI channel, use the controller `reset_scsi_channel` command.

### **Syntax**

```
controller reset_scsi_channel  
{integer}
```

### **Parameters**

`{integer}`

Specifies the controller channel (for example, 0, 1, 2, 3, etc.) on which you want to reset this SCSI channel. See the installation guide for your controller to determine the number of channels it actually supports.

### **Examples**

The following example resets the SCSI channel on channel 1:

```
AAC0>controller reset_scsi_channel 1
```

```
Executing: controller reset_scsi_channel 1
```

## **controller resume\_io**

To rescan the SCSI channels and resume all I/O activity on the currently opened, previously paused controller, use the `controller resume_io` command. You use this command after pausing the controller with the `controller pause_io` command.

### **Syntax**

```
controller resume_io
```

### **Examples**

The following example resumes all I/O activity on the currently paused controller:

```
AAC0> controller resume_io  
Executing: controller resume_io
```

### **Related Information**

controller commands:

- `controller pause_io` ([page 6-12](#))



## controller set automatic\_failover

Automatic failover allows you to replace a failed disk with a replacement disk. The controller then automatically assigns the disk you insert as a failover disk without your having to first assign it with the `array set failover` or `array set global_failover` command.

Note that the automatic failover feature works only with disks that reside in a SAF-TE or SES enclosure.

To turn on or off automatic failover for the specified controller, use the `controller set automatic_failover` command.




---

**Caution:** The controller deletes any data on the replacement disk when automatic failover is enabled and you remove the failed disk and insert the replacement disk in the failed disk's slot.

---

## Notes

A failover disk is a disk that you configure to automatically recover data from a failed partition/disk in a redundant array. To assign one or more failover disks for a single array, you use the `array set failover` command. To assign one or more failover disks for all arrays, you use the `array set global_failover` command. Although all array types accept failover assignments, only mirror sets and RAID-5 sets use the failover assignment if a disk fails. (Mirror sets and RAID-5 sets are often referred to as redundant arrays). The main characteristic of these commands is that you must “manually” assign the failover disk to one or more arrays. In fact, you might consider the previously described failover mechanism as “manual failover”.

The controller will assign a replacement drive as a failover disk only if the serial number of the drive is different from the serial number of the drive it replaces.

## Syntax

```
controller set automatic_failover
[/failover_enabled{=boolean}]
/failover_enabled{=boolean}
```

Specifies whether to turn on or off automatic failover on the specified controller. You must set this switch to `TRUE` if you want to turn on automatic failover on the specified controller.

To turn off automatic failover on the specified controller, set this switch to `FALSE`. The command defaults to `TRUE`, which means the command turns on automatic failover on the specified controller.

## Examples

Before turning the automatic failover feature on or off, check its status by using the `controller show automatic_failover` command.

The following example shows that the controller's automatic failover is currently off (disabled):

```
AAC0> controller show automatic_failover

Executing: controller show automatic_failover
Automatic failover DISABLED
```

The following example turns on (enables) the automatic failover feature for the currently opened controller:

```
AAC0> controller set automatic_failover /failover_enabled

Executing: controller set automatic_failover /failover_enabled=TRUE
```

The `controller show automatic_failover` command indicates that the controller's automatic failover is currently on (enabled):

```
AAC0> controller show automatic_failover

Executing: controller show automatic_failover
Automatic failover ENABLED
```

## Related Information

array commands:

- `array set failover` ([page 5-100](#))
- `array set global_failover` ([page 5-102](#))

controller commands:

- `controller show automatic_failover` ([page 6-22](#))

## controller set array\_verify

To enable or disable the array verify operation, use the `controller set verify` command. This command allows you to enable or disable the array verify operation for all arrays controlled by the specified controller.

### Syntax

```
controller set array_verify  
[/verify_enabled{=boolean}]  
/verify_enabled{=boolean}
```

Specifies whether to turn on or off the array verify feature on the specified controller. You must set this switch to `TRUE` if you want to turn on array verify for all arrays associated with the specified controller.

To turn off the array verify feature on the specified controller, set this switch to `FALSE`. The command defaults to `TRUE`, which means the command turns on the array verify feature on the specified controller.

### Examples

Before turning the array verify feature on or off, check its status by using the `controller show array_verify` command. The following example shows that the controller's array verify feature is currently off (disabled):

```
AAC0> controller show array_verify  
  
Executing: controller show array_verify  
Array verify DISABLED
```

The following example turns on (enables) the array verify feature for the currently opened controller:

```
AAC0> controller set array_verify /verify_enabled  
  
Executing: controller set array_verify /verify_enabled=TRUE
```

The `controller show array_verify` command indicates that the controller's array verify feature is currently on (enabled). The command also displays the number of errors (if any) found as a result of the array verify operation.

```
AAC0> controller show array_verify  
  
Executing: controller show array_verify  
Array verify ENABLED  
Errors found 0
```

## Related Information

controller commands:

- `controller show array_verify` ([page 6-25](#))

## **controller show automatic\_failover**

To display the automatic failover status (enabled or disabled) for the specified controller, use the `controller show automatic_failover` command.

### **Syntax**

```
controller show automatic_failover
```

### **Examples**

The following example shows that the controller's automatic failover is currently off (disabled):

```
AAC0> controller show automatic_failover  
  
Executing: controller show automatic_failover  
Automatic failover DISABLED
```

### **Related Information**

array commands:

- `array set failover` ([page 5-100](#))
- `array set global_failover` ([page 5-102](#))

controller commands:

- `controller set automatic_failover` ([page 6-18](#))

## controller show channels

To show all of the channels on a controller and the associated characteristics of each channel, use the `controller show channels` command. See the installation guide for your controller to determine the number of channels it actually supports.

### Syntax

```
controller show channels
```

### Examples

In the following example, the system displays the channels on the currently opened controller:

```
AAC0> controller show channels

Executing: controller show channels
Ch# Host ID Targets Type      Max Usage
---
0   15      15      NoInfo  NoInfo
```

The following sections provide more information on the items that the `controller show channels` command displays.

#### The Ch# and Host ID Items

The Ch# item displays the number of the channel on the controller. In the example there are four channels on the currently opened controller. See the installation guide for your controller to determine the number of channels it actually supports.

The Host ID item displays the device ID of the host controller, which in the example is 7 for all channels.

#### The Targets and Type Items

The Targets item displays the number of device IDs not including the controller ID, which in the example is 15 for all channels.

The Type item displays the SCSI channel type the channel is capable of supporting, which in the example is NoInfo for all channels.

### **The Max Usage Item**

The Max Usage item displays the best speed the channel is running at, which in the example is NoInfo for all channels. If drives other than Ultra-2 are present on the system, the entire channel is limited to UltraWide.

### **Related Information**

controller commands:

- controller details ([page 6-2](#))

## **controller show array\_verify**

To display the array verify status (enabled or disabled) for the specified controller, use the `controller show array_verify` command.

### **Syntax**

```
controller show array_verify
```

### **Examples**

Before turning the array verify feature on or off, check its status by using the `controller show array_verify` command.

The following example shows that the controller's automatic failover is currently on (enabled). The command also displays the number of errors (if any) found as a result of the array verify operation.

```
AAC0> controller show array_verify  
  
Executing: controller show array_verify  
Array verify ENABLED  
Errors found 0
```

### **Related Information**

controller commands:

- `controller set array_verify` ([page 6-20](#))



# 7

## disk Commands

### In this Chapter

<i>disk blink</i>	7-2
<i>disk initialize</i>	7-3
<i>disk list</i>	7-5
<i>disk remove dead_partitions</i>	7-9
<i>disk set default</i>	7-10
<i>disk set smart</i>	7-12
<i>disk show partition</i>	7-15
<i>disk show smart</i>	7-18
<i>disk show space</i>	7-22
<i>disk verify</i>	7-25
<i>disk zero</i>	7-27

Use the following syntax for disk commands:

```
disk command [object] [/switch{=value}]  
[parameter]
```

## disk blink

To cause a SCSI disk access light to blink (or stop blinking), use the `disk blink` command.

### Syntax

```
disk blink [/wait{=boolean}] {scsi_device}
{integer}
```

### Parameters

{scsi\_device}

Specifies the ID for the SCSI disk you want to blink. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), device ID (0 through 15 inclusive), and device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

{integer}

Specifies the number of seconds you want the SCSI disk to blink. A value of zero (0) stops the SCSI disk from blinking.

### Switches

/wait{=boolean}

Specifies whether to perform verification synchronously or asynchronously. If you set this switch to `TRUE`, the command performs the block verification synchronously and the command prompt does not return until the block verification operation completes.

The default is `FALSE`; the command performs the block verification asynchronously and the command prompt returns immediately.

### Examples

The following example causes SCSI channel number 1, device ID 2, and device logical unit number 0 to blink for fifty-five seconds:

```
AAC0>disk blink (1,2,0) 55
Executing: disk blink (CHANNEL=1,ID=2,LUN=0) 55
```

## disk initialize

To initialize a SCSI disk for use with the currently opened controller, use the `disk initialize` command. This command writes data structures to the disk so that the controller can use the disk.

### Syntax

```
disk initialize [/always{=boolean}]
[/unconditional{=boolean}]{scsi_device}
```

### Parameters

```
{scsi_device}
```

Specifies the ID for the SCSI disk you want to prepare and initialize for controller use. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), device ID (0 through 15 inclusive), and device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

### Switches

```
/always{=boolean}
```

Specifies whether to initialize the disk even if the disk has data on it. If you specify `TRUE` for this switch, the command initializes the disk even if it has data on it. If arrays on the disk have open files, this command cannot be used. In other words, you need to close all open files before you use this command to initialize a SCSI disk.

The default behavior for the command is `FALSE`; that is, the `disk initialize` command does not initialize a disk that has data on it. Specifying `/always` overrides this behavior.

```
/unconditional{=boolean}
```

Specifies whether to initialize the disk even if arrays on the disk have open files. If you specify `TRUE` for this switch, the command initializes the disk even if the disk has open files.

The default is `FALSE`; that is, the `disk initialize` command does not initialize a disk that has open files on it.



---

**WARNING:** Unconditionally deleting an array that is in use can cause a system crash under some circumstances.

---

## Examples

The following example initializes a SCSI disk device (SCSI channel number 0, device ID 2, and device logical unit number 0) that has data on it:

```
AAC0>disk initialize /always (0,2,0)
```

```
Executing: disk initialize /always=TRUE (CHANNEL=0,ID=2,LUN=0)
```

## disk list

To display a list of the disks available on the currently opened controller, use the `disk list` command.

### Syntax

```
disk list [/all{=boolean}] [/full{=boolean}]  
[{scsi_device}]
```

### Parameters

{scsi\_device}

Specifies the ID for a specific device for which you want to display information. A device consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), device ID (0 through 15 inclusive), and device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

### Switches

/all{=boolean}

Specifies whether to display a list of all devices. If you set this switch to `TRUE`, the command displays a list of all devices on the controller.

This switch defaults to `FALSE` if you specify `scsi_device`. Otherwise, the switch defaults to `TRUE`.

/full{=boolean}

Specifies whether to display detailed information. If you specify `TRUE`, the command displays detailed information. If you specify `FALSE`, the command does not display detailed information.

This switch defaults to `FALSE`.

### Examples

The following example shows how to display nondetailed information for all of the SCSI disks on the currently opened controller:

```
AAC0>disk list
```

```

Executing: disk list
C:ID:L  Device Type Blocks      Bytes/Block  Usage      Shared Rate
-----
2:00:0  Disk          8887200     512         Initialized NO    132
2:01:0  Disk          8887200     512         Initialized NO    132
2:02:0  Disk          8496960     512         Initialized NO    132
2:03:0  Disk          8887200     512         Initialized NO    132

```

The detailed `disk list` display contains the previous columns of information plus the following columns:

- Removable media
- Vendor-ID
- Product-ID
- Rev

The following sections provide more information on the columns of information that the `disk list` command displays.

#### The C:ID:L Column

The **C:ID:L** column displays the device channel number, the device ID, and the device logical unit number for each disk on the currently opened controller. (The display shows only one disk if you specified a particular disk.)

In the example, the command displays the SCSI channel number, the device ID, and the device logical unit number for the four disks on the open controller.

#### The Device Type Column

The **Device Type** column displays the device type. The `disk list` command can display one of the values listed in Table 7-1.

Table 7-1. Device Type Values

Value	Meaning
Disk	The device type is a SCSI direct access device. Typically, these are disk drives. This value appears in the example.
Sequential	The device type is a SCSI sequential access device; typically, tape drives.
Printer	The device type is a SCSI printer device.

Value	Meaning
Processor	The device type is a processor device.
Write once	The device type is a SCSI write once and read many times device. Typically, these devices are referred to as WORM disk drives.
CDROM	The device type is a SCSI CD-ROM (read-only direct access device).
Scanner	The device type is a SCSI scanner device.
Optical	The device type is a SCSI optical disk device.
Medium changer	The device type is a SCSI medium changer device; typically, a changer for a Jukebox.
Network	The device type is a SCSI network communication device.

#### The Removable media Column

The **Removable media** column displays the value **Y** if the media in the device is removable and the value **N** if the media in the device is not removable.

#### The Vendor-ID and Product-ID Columns

The **Vendor-ID** column displays a string of characters that identifies the vendor of the device.

The **Product-ID** column displays a string of characters that identifies the product line associated with the device.

#### The Rev Column

The **Rev** column displays the revision number of the device.

#### The Block Column

The **Block** column displays the number of blocks available on the device.

#### The Bytes/Block Column

The **Bytes/Block** column displays the number of bytes for each block on the device.

### The Usage Column

The **Usage** column displays the usage of the device. The `disk list` command can display one of the values listed in Table 7-2.

Table 7-2. SCSI Disk Device Usage Values

Value	Meaning
Detached	The system detected that the diskset in the detached state. When a diskset is detached, it is not available for use.
DOS	The device was prepared for use (initialized) with MS-DOS partitions.
Initialized	The device was prepared for use (initialized) with arrays.
Not Initialized	The device was not prepared for use (initialized) with arrays.
Offline	The device was present at boot time. However, the device is either removed from the controller or it failed.
Unowned	The controller does not own the device.

### The Shared Column

The **Shared** column displays the value **YES** if the device resides on a shared channel and the value **NO** if the device does not reside on a shared channel. In the example, the devices do not reside on a shared channel.

### The Rate Column

The **Rate** column displays the negotiated speed of the device, in megabytes per second.

### Related Information

disk commands:

- `disk initialize` ([page 7-3](#))
- `disk show partition` ([page 7-15](#))
- `disk show space` ([page 7-22](#))



## disk remove dead\_partitions

To remove all dead partitions from a SCSI disk, use the `disk remove dead_partitions` command. A dead partition is a partition that is no longer used by any array.

Typically, you use this command only under specific circumstances. For example, if you remove a disk from a redundant array and then later add the disk to the controller, the partition on the reinserted disk (which was previously part of the redundant array) is no longer useful. In this case, you use this command and specify the disk that was added to the controller again.

If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column (displayed with the `array list` command) changes to a “!” (exclamation point).

### Syntax

```
disk remove dead_partitions {scsi_device}
```

### Parameters

```
{scsi_device}
```

Specifies the ID for the SCSI disk from which you want to remove all dead partitions. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), device ID (0 through 15 inclusive), and device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

### Examples

The following example removes all dead partitions that reside on a SCSI disk device (SCSI channel number 0, device ID 2, and device logical unit number 0):

```
AAC0>disk remove dead_partitions (0,02,0)
```

```
Executing: disk remove dead_partitions (CHANNEL=0,ID=2,LUN=0)
```

## **Related Information**

array commands:

- array list ([page 5-54](#))

## disk set default

To set the default SCSI ID for use in subsequent CLI commands, use the `disk set default` command. This command allows you to set defaults for a specific device's channel number and device logical unit number. Then, in subsequent commands, you need only specify the device ID on the command line.

### Syntax

```
disk set default {scsi_device}
```

### Parameters

```
{scsi_device}
```

Specifies the ID for the device. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), a device ID (0 through 15 inclusive), and a device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

If you do not specify a device ID, the command displays an appropriate error message.

### Examples

The following example sets the SCSI channel number and SCSI default logical unit number for device (0,3,0) as the default for use in subsequent CLI commands:

```
AAC0>disk set default (1,3,0)
```

```
Executing: disk set default (CHANNEL=1,ID=3,LUN=0)
```

You need only specify the device ID in subsequent CLI commands, and the CLI assumes the default SCSI channel number and SCSI logical unit number as follows:

```
AAC0>disk list 3
```

```
Executing: disk list (ID=3)
```

C:ID:L	Device	Type	Blocks	Bytes/Block	Usage	Shared
2:03:0	Disk		4197405	512	Initialized	NO

See the `disk list` command for information on the items in the display.

## **Related Information**

disk commands:

- disk list ([page 7-5](#))

## disk set smart

To change a device's S.M.A.R.T. configuration, use the `disk set smart` command. The acronym S.M.A.R.T. stands for Self-Monitoring, Analysis and Reporting Technology. This technology is an industry standard for hard drives that monitors a variety of disk parameters, such as the rate of read-write errors. In addition, S.M.A.R.T. can send an alert to system administrators about potential problems caused by disk errors.

This command allows you to make the following changes related to S.M.A.R.T.:

- Enable or disable S.M.A.R.T. configuration for all disks on the system
- Clear the S.M.A.R.T. error count for the specified disk
- Enable or disable S.M.A.R.T. exception reporting

### Syntax

```
disk set smart [/all{=boolean}][ /clear{=boolean}]
[/enable_exceptions{=boolean}][ /logerr{=boolean}]
[/perf{=boolean}][ /report_count{=integer}]
[/update{=boolean}] [{scsi_device}]
```

### Parameters

{scsi\_device}

Specifies the ID for the SCSI disk on which you want to change S.M.A.R.T. configurations. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), device ID (0 through 15 inclusive), and device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

You do not need to specify a SCSI ID if you use the `/all` switch.

## Switches

`/all{=boolean}`

Specifies whether to enable S.M.A.R.T. configurations (enable exception reporting) for all disks on the system. If you specify `TRUE` for this switch, the command enables S.M.A.R.T. configurations for all disks on the system.

The default is `FALSE`; the `disk set smart` command does not enable S.M.A.R.T. configurations for all disks on the system. In this case, you would want to specify a SCIS ID to enable S.M.A.R.T. configurations for a specific disk on the system.

`/clear{=boolean}`

Specifies whether to clear S.M.A.R.T. error counts for the specified disk or disks. If you specify `TRUE` for this switch, the command clears error counts for the specified disk or disks.

The default is `FALSE`; that is, the `disk set smart` command does not clear S.M.A.R.T. error counts for disks on the system.

`/enable_exceptions{=boolean}`

Specifies whether to enable S.M.A.R.T. exceptions reporting for the specified disk or disks. If you specify `TRUE` for this switch, the command enables exception reporting for the specified disk or disks.

The default is `FALSE`; that is, the `disk set smart` command does not enable S.M.A.R.T. exception reporting for disks on the system.

`/interval_timer{=integer} <?>`

`/logerr{=boolean}`

Specifies whether to enable logging of S.M.A.R.T. exception reporting for the specified disk or disks. If you specify `TRUE` for this switch, the command enables logging of S.M.A.R.T. exception reporting for the specified disk or disks.

If you specify `FALSE` for this switch, the command disables logging of S.M.A.R.T. exception reporting for the specified disk or disks.

```
/mrie={integer} <?>
```

```
/perf{=boolean}
```

Specifies whether to report exceptions according to the MRIE mode taking into account performance. If you specify `TRUE` for this switch, the command performs exception reporting as long as performance is not an issue. If performance is an issue, the command does not report exceptions.

If you specify `FALSE` for this switch, exceptions are reported according to the MRIE mode regardless of performance issues.

```
/report_count{=integer}
```

Specifies the number of times an exception can be reported. The value 0 (zero) indicates that there is no limit to the number of times an exception can be reported.

```
/update{=boolean}
```

Specifies whether to update the number of device errors found on the specified device. If you specify `TRUE` for this switch, the command updates the number of device errors found on the specified device.

If you specify `FALSE` for this switch, the command does not update the number of device errors found on the specified device.

## Examples

The following example enables S.M.A.R.T. configurations on one SCSI disk device (SCSI channel number 1, device ID 2, and device logical unit number 0):

```
AAC0>disk set smart (1,2,0)
```

```
Executing: disk set smart (CHANNEL=1,ID=2,LUN=0)
```

## Related Information

disk commands:

- disk show smart ([page 7-18](#))

## disk show partition

To display a list of partitions on the disks attached to the currently opened controller, use the `disk show partition` command.

### Syntax

```
disk show partition
```

### Examples

The following example shows a sample output from the `disk show partition` command:

```
AAC0>disk show partition

Executing: disk show partition
Scsi   Partition   Array   MultiLevel
C:ID:L Offset:Size  Num Type   Num Type   R/W
-----
0:02:0 64.0KB:20.0MB 0   Volume 0   None      RW
```

The following sections provide more information on the columns that the `disk show partition details` command displays.

#### The SCSI C:ID:L Columns

The **C:ID:L** column displays the SCSI channel number, the device ID, and the device logical unit number for each disk on the currently opened controller.

In the example, the command displays the SCSI channel number, the device ID, and the device logical unit number for one disk (0:02:0) on the currently opened controller.

#### The Partition Offset:Size Column

The **Partition Offset:Size** column displays the offset (in bytes) into the device and the size of the partition (in bytes). In the example, the offset is **64.0 KB** and the size is **20.0 MB**.

If a partition is dead, the “:” (colon) in the **Partition Offset:Size** column changes to a “!” (exclamation point). See the `disk remove dead_partitions` ([page 7-9](#)) command for more information on dead partitions.



### The Array Num and Array Type Columns

The **Array Num** column displays the ID of the primary array in the partition. In the example, the array ID is **0**.

The **Array Type** column displays the type of array in which the partition resides. Table 7-3 lists the possible values that the command can display in the Type column.

Table 7-3. Primary Array Values

Value	Meaning
None	This partition is not in an array.
Mirror	This partition is part of a mirror set.
Stripe	This partition is part of a stripe set.
Volume	This partition is part of a volume set.
RAID-5	This partition is part of a RAID-5 set.
Reconf	This partition is part of an array reconfiguration operation. An array reconfiguration operation occurs when you use the <code>array reconfigure</code> command to change an array from one type to another. The Reconf value is not used in UNIX.

### The MultiLevel Num and MultiLevel Type Columns

The **MultiLevel Num** column displays the ID of the multilevel array the partition is associated with (if any). In the example, the array ID is **0**.

The **MultiLevel Type** column displays the type of multilevel array in which the partition resides. Table 7-4 lists the possible values that the command can display in the Type column.

Table 7-4. Multilevel Array Values

Value	Meaning
None	This partition is not in a secondary array.
Stripe	This partition is part of a stripe set.
Volume	This partition is part of a volume set.

**The R/W Column**

The **R/W** column displays whether the partition is read-only or read-write. Specifically, the column can display the values listed in Table 7-5.

Table 7-5. R/W Array Values

Value	Meaning
RO	The partition is read-only.
RW	The partition is read-write.

## disk show smart

To display S.M.A.R.T. configuration information for one or all disks, use the `disk show smart` command. The acronym S.M.A.R.T. stands for Self-Monitoring, Analysis and Reporting Technology. This technology is an industry standard for hard drives that monitors a variety of disk parameters, such as the rate of read-write errors. In addition, S.M.A.R.T. can send an alert to system administrators about potential problems caused by disk errors.

### Syntax

```
disk show smart [/all{=boolean}] [/full{=boolean}]
[/view_changeable{=boolean}][{scsi_device}]
```

### Parameters

```
{scsi_device}
```

Specifies the ID for the SCSI disk for which you want to display S.M.A.R.T. configuration information. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), device ID (0 through 15 inclusive), and device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

You do not need to specify a SCSI ID if you use the `/all` switch.

### Switches

```
/all{=boolean}
```

Specifies whether to display S.M.A.R.T. configuration information for all disks on the system. If you specify `TRUE` for this switch, the command displays S.M.A.R.T. configuration information for all disks on the system.

This switch defaults to `TRUE`.

```
/full{=boolean}
```

Specifies whether to display detailed S.M.A.R.T. configuration information for one or all SCSI disks. If you specify `TRUE`, the command displays detailed S.M.A.R.T. configuration

information. If you specify `FALSE`, the command does not display detailed S.M.A.R.T. configuration information.

This switch defaults to `FALSE`.

`/view_changeable{=boolean}`

Specifies whether to display the configuration information that can be set on a S.M.A.R.T. disk. If you specify `TRUE` for this switch, the command displays the letter “X” in the column for the configuration information that can be set. For example, an “X” appears in the Enable Exception Control column for those S.M.A.R.T. disks on which this value can be set.

This switch defaults to `FALSE`.

## Examples

The following example shows how to display nondetailed S.M.A.R.T. configuration information for all disks on the system:

```
AAC0>disk show smart
```

```
Executing: disk show smart
```

	Smart Capable	Method of Informational Exceptions (MRIE)	Enable Exception Control	Performance Enabled	Error Count
C:ID:L	Device	-----	-----	-----	----
3:00:0	N				
3:01:0	N				
3:02:0	N				
3:03:0	N				

The detailed `disk show smart` display contains the previous columns of information plus the following columns:

- Log Errors
- Interval Timer (secs.)
- Report Count

The following sections provide more information on the columns that the `disk show smart` command displays.

### The C:ID:L Column

The **C:ID:L** column displays the SCSI channel number, the device ID, and the device logical unit number for each disk on the currently

opened controller. (The display shows only one disk if you specified a particular disk.)

In the example, the command displays the SCSI channel number, the device ID, and the device logical unit number for the four disks on the open controller.

#### **The Smart Capable Device Column**

The **Smart Capable Device** column displays the value **Y** if the device is S.M.A.R.T. capable or the value **N** if the device is not S.M.A.R.T. capable.

#### **The Method of Informational Exceptions (MRIE) Column <?>**

The **Method of Informational Exceptions (MRIE)** column displays the MRIE value (one of a possible six values). However, you cannot set the MRIE. Your devices report exceptions only when polled (level 6) and you cannot change the setting for this switch.

#### **The Enable Exception Control Column**

The **Enable Exception Control** column displays the value **Y** if you enabled S.M.A.R.T. exception control reporting or the value **N** if you did not enable S.M.A.R.T. exception control reporting.

If you specified the `/view_changeable` switch, the command displays an **X** in this column for those disks on which you can enable S.M.A.R.T. exception control reporting.

#### **The Performance Enabled Column**

The **Performance Enabled** column displays the value **Y** if you enabled performance or **N** if you did not enable performance. You enable performance by using the `disk set smart` command's `/perf` switch.

If you specified the `/view_changeable` switch, the command displays an **X** in this column for those disks on which you can enable performance.

#### **The Log Errors Column**

The **Log Errors** column displays the value **Y** if you enabled log errors or **N** if you did not enable log errors. You enable log errors by using the `disk set smart` command's `/logerr` switch.

If you specified the `/view_changeable` switch, the command displays an **X** in this column for those disks on which you can enable log errors.

#### **The Interval Timer (secs.) Column**

The **Interval Timer (secs.)** column displays the number of seconds specified for the exception reporting interval. If you specified the `/view_changeable` switch, the command displays an **X** in this column for those disks on which you can set the number of seconds for the exception reporting interval.

#### **The Report Count Column**

The **Report Count** column displays the number of times you specified for exceptions to be reported. You specify this number by using the `disk set smart` command's `/report_count` switch.

#### **The Error Count Column**

The **Error Count** column displays the number of errors that S.M.A.R.T. encountered on the disk.

If you specified the `/view_changeable` switch, the command displays an **X** in this column for those disks on which you can enable the recording of errors that S.M.A.R.T. encounters on the disk.

### **Related Information**

disk commands:

- `disk set smart` ([page 7-12](#))

## disk show space

To display space usage information on a SCSI disk, use the `disk show space` command.

### Syntax

```
disk show space [/all{=boolean}] [{scsi_device}]
```

### Parameters

{scsi\_device}

Specifies the ID for the device for which you want to display space usage information. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), a device ID (0 through 15 inclusive), and a device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

### Switches

/all{=boolean}

Specifies whether to show space usage information for all SCSI disks on the currently opened controller. If you set this switch to `TRUE`, the command shows space usage information for all SCSI disks on the currently opened controller.

This switch defaults to `FALSE` if you specify the `scsi_device` parameter; otherwise, the switch defaults to `TRUE`.

### Examples

The following example shows space usage information for all of the SCSI disks on the currently opened controller:

```
AAC0>disk show space
```

```
Executing: disk show space
```

Scsi C:ID:L	Usage	Size
0:02:0	Array	64.0KB: 20.0MB
0:02:0	Free	20.0MB: 29.0MB
0:03:0	Free	64.0KB: 49.0MB
0:04:0	Free	64.0KB: 49.0MB
0:05:0	Free	64.0KB: 49.0MB

The following example shows space usage information for SCSI disk device (0,2,0):

```
AAC0>disk show space (0,2,0)

Executing: disk show space (ID=2)
Scsi C:ID:L Usage      Size
-----
0:02:0   Array      64.0KB:  20.0MB
0:02:0   Free       20.0MB:  29.0MB
```

The following sections provide more information on the columns that the `disk show space` command displays.

### The Scsi C:ID:L Column

The **Scsi C:ID:L** column displays the SCSI channel number, the device ID, and the device logical unit number for each disk on the currently opened controller. (The display shows only one disk if you specified a particular disk.)

In the first example, the command displays the SCSI channel number, the device ID, and the device logical unit number for the six disks on the open controller. In the second example, the command displays information for the specified device (0:02:0).

### The Usage Column

The **Usage** column displays the type of partition or some characteristic related to disk space usage. Specifically, the command can display the values listed in Table 7-6.

Table 7-6. Disk Usage Values

Value	Meaning
Free	This area is unused space on the disk. In the example, a specific area is unused space on the 0:02:0 disk. This unused space is typically referred to as freespace.
Array	This partition is used as part of an array. In the example, a partition on the 0:02:0 disk is used as part of an array.
Orphan	This area was part of an array that could not be configured.
Dead	This area had an error and was declared dead.
Conflict	This area is not configured because of a conflict.



### The Size Column

The **Size** column displays the offset (in bytes) into the device and the size of the partition or space (in bytes). In the example, the offset and sizes for the first two disks are as follows:

- For the 0:02:0 disk

The offset is **64.0 KB** for the partition and **20.0 MB** for the unused space. The size is **20.0 MB** for the partition and **29.0 MB** for the unused space.

- For the 0:03:0 disk

The offset is **64.0 KB** and the size is **49.0 MB** for the unused space. (This disk has no partitions on it.)

The offset is **64.0 KB** and the size is **49.0 MB** for the unused space. (This disk has no partitions on it.)

## disk verify

To verify all blocks on a SCSI disk device and, optionally, repair any bad blocks, use the `disk verify` command.

### Syntax

```
disk verify [/repair{=boolean}] [/wait{=boolean}]
{scsi_device}
```

### Parameters

```
{scsi_device}
```

Specifies the ID for the SCSI disk device that you want to verify. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), a device ID (0 through 15 inclusive), and a SCSI logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

### Switches

```
/repair{=boolean}
```

Specifies whether to automatically repair bad blocks. If you set this switch to `TRUE`, the command attempts to repair any bad blocks. If you do not specify the switch, the command only reports failures.

This switch defaults to `FALSE`.

```
/wait{=boolean}
```

Specifies whether to perform verification synchronously or asynchronously. If you set this switch to `TRUE`, the command performs the block verification synchronously and the command prompt does not return until the block verification operation completes.

The default is `FALSE`; that is, the command performs the block verification asynchronously and the command prompt returns immediately.

## Examples

The following example synchronously verifies and repairs all blocks on disk (0,2,0):

```
AAC0>disk verify /repair=TRUE /wait=TRUE (0,2,0)
```

```
Executing: disk verify (CHANNEL=0,ID=2,LUN=0)
```

As the command executes, note the title bar of the DOS window displays the status of the command. For example:

```
Stat:OK!Task:100,Func:SCV,State:RUN 97.2%
```

See [page 3-9](#) for detailed information on the status information that the command displays in the title bar as it executes the task.

## disk zero

To clear an entire SCSI disk, use the `disk zero` command. When you clear a disk, all data is erased and cannot be recovered.

### Command And Switch Availability

This command and its associated switch (or switches) are supported on the following operating systems:

- Windows NT
- Windows 2000 (basic disk only)
- Windows XP (basic disk only)
- NetWare

### Syntax

```
disk zero [/always{=boolean}] [/wait{=boolean}]
{scsi_device}
```

### Parameters

```
{scsi_device}
```

Specifies the ID for the SCSI disk you want to clear. A SCSI ID consists of a SCSI channel number (for example, 0, 1, 2, 3, etc.), device ID (0 through 15 inclusive), and device logical unit number (0 through 7 inclusive). See the `scsi_device` section on page 3-7 for more information on how to specify the SCSI device.

### Switches

```
/always{=boolean}
```

Specifies whether to clear the disk, even if it has data on it. If you specify `TRUE`, the command clears the disk even if it has data on it.

The default behavior for the command is `FALSE`; the command clears the disk only if it has no data on it. Specifying `/always` overrides this behavior.

In both cases, all user files must be closed. The `/always` switch cannot override this restriction.

`/wait{=boolean}`

Specifies whether to clear the disk synchronously or asynchronously. If you set this switch to `TRUE`, the command clears the disk synchronously and the command prompt does not return until the clear disk task completes.

The default is `FALSE`; the command clears the disk asynchronously and the command prompt returns immediately.

## Examples

The following example clears SCSI disk (0,2,0). There is data on the disk. However, the data is not deemed important and thus the `/always` switch is used:

```
AAC0> disk zero /always /wait (0,2,0)
```

```
Executing: disk zero /always=TRUE /wait=TRUE (CHANNEL=0,ID=2,LUN=0)
```

As the command executes, note the title bar of the DOS window displays the status of the command. For example:

```
Stat:OK!Task:100,Func:SCZ,State:RUN 97.2%
```

See [page 3-10](#) for detailed information on the status information that the command displays in the title bar as it executes the task.

# 8

## logfile Commands

In this Chapter

<i>logfile end</i>	8-2
<i>logfile start</i>	8-3

Use the following syntax for logfile commands:

```
logfile command [/switch{=value}] [parameter]
```

## logfile end

To end the logging of all output, use the `logfile end` command.

### Syntax

```
logfile end
```

### Examples

The following example stops logging information to the file `aac0log17Nov01.txt` (which was previously started with the `logfile start` command):

```
AAC0> logfile end
```

```
Executing: logfile end
```

```
Log file closed.
```

### Related Information

logfile commands:

- `logfile start` ([page 8-3](#))

## logfile start

To begin the logging of all CLI command line activity to a specified file, use the `logfile start` command. CLI command line activity includes the associated output the CLI command displays.

The CLI continues to log command output until you close the CLI or you explicitly end the logfile with the `logfile end` command.

### Syntax

```
logfile start [/append{=boolean}] {string}
```

### Parameter

{string}

Specifies the name of the file to contain CLI command line activity. On Windows, use the Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only) file name conventions. If the file name has spaces, use quotation marks (“”) to enclose the string.

On NetWare, use the NetWare file name conventions.

On UNIX, use the UNIX file name conventions.

### Switches

/append{=boolean}

Specifies whether to append all CLI command line activity to an existing output file, if one exists. If you set this switch to `TRUE`, the command appends the CLI command line activity to the file specified in {string}.

This switch defaults to `FALSE` (that is, the command does not append CLI command line activity to an existing output file).

### Examples

The following example (Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only)) logs all output to a file called `c:\ctrloginfo\aac0log17Nov01.txt`.



The .txt extension allows the file to be read by the Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only) Notepad application:

```
AAC0> logfile start c:\ctrloginfo\aac0log17Nov01.txt  
Executing: logfile start "c:\ctrloginfo\aac0log17Nov01.txt"  
File c:\ctrloginfo\aac0log17Nov01.txt receiving all output.
```

The following example (Windows NT/Windows 2000 (basic disk only)/Windows XP (basic disk only)) logs subsequent output to an existing file:

```
AAC0> logfile start /append c:\ctrloginfo\aac0log17Nov01.txt  
Executing: logfile start /append=TRUE  
"c:\ctrloginfo\aac0log17Nov01.txt"  
  
File c:\ctrloginfo\aac0log17Nov01.txt receiving all output.
```

## Related Information

logfile commands:

- logfile end ([page 8-2](#))

# 9

## task Commands

### In this Chapter

<i>task list</i>	9-2
<i>task resume</i>	9-8
<i>task stop</i>	9-10
<i>task suspend</i>	9-12

Use the following syntax for task commands:

```
task command [/switch{=value}] [parameter]
```

## task list

To display a list of one or all tasks running on the currently opened controller, use the `task list` command. A task is an operation that occurs only on the controller, asynchronous to all other operations. Clearing a disk, creating a file system, and creating a mirror set are examples of tasks done on the controller.

The command displays an appropriate message if there are no tasks running on the controller.

### Syntax

```
task list [/all{=boolean}]
```

### Parameters

```
{integer}
```

Specifies the task ID for the task whose information you want to display. You need to perform a `task list /all` to display all tasks and their associated IDs.

### Switches

```
/all{=boolean}
```

Specifies whether to display all currently running tasks. If you set this switch to `TRUE`, the command displays a list of all tasks running on the currently opened controller.

This switch defaults to `FALSE` if you specify the `integer` parameter; otherwise, the switch defaults to `TRUE`.

### Examples

The following example shows typical output as a result of using the `task list` command.

```
AAC0> task list

Executing: task list
Controller Tasks
TaskId  Function  Done%  Array      State  Specific1 Specific2
-----  -
101     Scrub     24.6%  0           SUS    00000000  00000000
```

As the example shows, the typical display shows the following columns of information:

- TaskId
- Function
- Done%
- Array
- State
- Specific1
- Specific2

The following sections describe each column that the `task list` command displays.

#### The Task ID Column

The **Task ID** column displays the ID number associated with a specific task. The controller software assigns each task a unique ID number.

#### The Function Column

The **Function** column displays the type of task running on the controller. Table 9-1 describes the types of tasks that the `task list` command can display.

Table 9-1. Task Types

Task Type	Meaning
Create	A create mirror set or create multilevel mirror set task is running on the specified array. When the create mirror set or create multilevel mirror set task completes, the specified array is a mirror set or a multilevel array of mirror sets. The create a mirror set or create a multilevel mirror set task runs as a result of using the <code>array create mirror</code> or <code>array create mmirror</code> command.
FATfmt	An array format FAT file system task is running on the specified array. The format FAT file system task runs as a result of using the <code>array format</code> command with the <code>/ file_system</code> switch set to FAT.

Table 9-1. Task Types (Continued)

Task Type	Meaning
NTFSfmt	An array format NTFS file system task is running on the specified array. The format NTFS file system task runs as a result of using the <code>array format</code> command with the <code>/file_system</code> switch set to NTFS.
Rebuild	A rebuild task is running on the specified array. Typically, the rebuild task runs when the controller is in a rebuild of a redundant array (RAID-5 set, mirror set, or multilevel array of mirror sets).
Reconfg	An array reconfigure task is running on the specified array. When the array reconfigure task completes, the specified array becomes a different array (for example, from a volume set to a stripe set). The array reconfigure task runs as a result of using the <code>array reconfigure</code> command.
Scrub	A scrub task is running on the specified redundant array. When the scrub task completes, the specified redundant array has reconstructed data on one partition based on data found on the other partition (for mirror sets and multilevel arrays of mirror sets). Or, the specified redundant array recalculates and replaces, if necessary, the parity information (for RAID-5 sets). The scrub task runs as a result of the <code>array scrub</code> command. The scrub task also runs as a result of the <code>array create raid5</code> command with the <code>/scrub</code> switch specified. This is the task type displayed in the example.
Unknown	An unknown task is running.

Table 9-1. Task Types (Continued)

Task Type	Meaning
Verify	<p>A verify with no repair of bad blocks task is running on the specified SCSI disk. When the verify with no repair of bad blocks task completes, the specified SCSI disk's blocks were verified without repairing any detected defects. The verify with no repair of bad blocks task runs as a result of using the <code>disk verify</code> command without specifying the <code>/repair</code> switch.</p> <p>Or, a verify with repair of bad blocks task is running on the specified SCSI disk. When the verify with repair of bad blocks task completes, the specified SCSI disk's blocks were verified and any detected bad blocks were repaired. The verify with repair of bad blocks task runs as a result of using the <code>disk verify</code> command with the <code>/repair</code> switch.</p>
Zero	<p>A clear disk task is running on the specified SCSI disk. When the clear disk task completes, the specified SCSI disk is cleared (that is, all data is erased and cannot be recovered). The clear disk task runs as a result of using the <code>disk zero</code> command.</p>

### The Done% Column

The **Done%** column displays a running percentage value that indicates the progress of the currently running task. The display shows the percentage in tenths of a percent increments. The currently running task is complete when the **Done %** column displays **100%**.

### The Array Column

The **Array** column displays the ID number of the array associated with the task (if any). In the example, the array's ID is **0** (zero).

### The State Column

The **State** Column displays the state of the task. Table 9-2 describes the state values that the `task list` command can display.

Table 9-2. State Values

State	Meaning
BAD	The task failed and is no longer running.
DNE	The task successfully completed or the task is no longer running (that is, the task is done).
RUN	The task is running.
SUS	The task was suspended. Typically, you suspend a task with the <code>task suspend</code> command.
UNK	The controller reported an unknown status for the task.

### The Specific1 Column

The **Specific1** column displays additional information for certain tasks that appear in the **Task Id** column. Table 9-3 lists the tasks for which the `task list` command can display additional information. Table 9-3 also describes the additional information that `task list` can display.

Table 9-3. Additional Specific1 Information

Task	Additional Information
Verify. Refers to the verify with no repair of bad blocks task, which runs as a result of using the <code>disk verify</code> command without the <code>/repair</code> switch.	Displays the number of bad blocks found with no replacement. The example displays zeroes, which signifies no errors found.
Scrub. Refers to the scrub task, which runs as a result of using the <code>array scrub</code> command.	Displays the number of sectors that were different.

### The Specific2 Column

The **Specific2** column displays additional information for certain tasks that appear in the **Task Id** column. Table 9-4 lists the tasks for which the `task list` command can display additional information. Table 9-4 also describes the additional information that `task list` can display.

Table 9-4. Additional Specific2 Information

Task	Additional Information
Verify. Refers to the verify with repair of bad blocks task, which runs as a result of using the disk verify command with the / repair switch.	Displays the number of bad blocks found that were replaced.

## Related Information

array commands:

- array create mirror ([page 5-5](#))
- array scrub ([page 5-94](#))

disk commands:

- disk verify ([page 7-26](#))
- disk zero ([page 7-28](#))

task commands:

- task resume ([page 9-8](#))
- task stop ([page 9-10](#))
- task suspend ([page 9-12](#))



## task resume

To resume one or all tasks previously suspended on the currently opened controller, use the `task resume` command. You suspend tasks with the `task suspend` command.

### Syntax

```
task resume [/all{=boolean}]
[integer]
```

### Parameters

{integer}

Specifies the task ID for the task to resume. Perform a `task list /all` to display the task IDs for all tasks currently suspended on the open controller before resuming a specific task or all tasks.

### Switches

/all{=boolean}

Specifies whether to resume all suspended tasks. If you set this switch to `TRUE`, the command resumes all tasks that were suspended on the currently opened controller.

If you do not specify this switch or a valid task ID, the command displays an appropriate error message.

### Examples

Before resuming a suspended task, use the `task list` command to identify any suspended tasks, as in the following example.

```
AAC0>task list
```

```
Executing: task list
```

TaskId	Function	Done%	Array	State	Specific1	Specific2
101	Scrub	24.6%	0	SUS	00000000	00000000

The following example shows the use of the `task resume` command to resume task 101, which was previously stopped with the `task suspend` command:

```
AAC0>task resume 101
```

```
Executing: task resume 101
```

Use the `task list` command after using the `task resume` command to display information about the task, as in the following example:

```
AAC0>task list
```

```
Executing: task list
```

TaskId	Function	Done%	Array	State	Specific1	Specific2
101	Scrub	24.6%	0	RUN	00000000	00000000

As a result of using the `task resume` command, the `State` column now displays `RUN` (running on the controller) instead of `SUS` (suspended on the controller).

## Related Information

`task` commands:

- `task list` ([page 9-2](#))
- `task suspend` ([page 9-12](#))

## task stop

To stop one or all tasks running on the currently opened controller, use the `task stop` command.

### Syntax

```
task stop [/all{=boolean}]
```

### Parameters

```
{integer}
```

Specifies the task ID for the task to stop. Perform a `task list /all` to display the task IDs for all tasks currently running on the open controller before stopping a specific task or all tasks.

### Switches

```
/all{=boolean}
```

Specifies whether to stop all running tasks. If you set this switch to `TRUE`, the command stops all tasks that were running on the currently opened controller.

If you do not specify this switch or a valid task ID, the command displays an appropriate error message.

### Examples

Before stopping a running task, use the `task list` command to identify any currently running tasks, as in the following example:

```
AAC0>task list
```

```
Executing: task list
```

TaskId	Function	Done%	Array	State	Specific1	Specific2
101	Scrub	24.6%	0	RUN	00000000	00000000

The following example shows the use of the `task stop` command to stop task 101:

```
AAC0> task stop 101
```

```
Executing: task stop 101
```

Use the `task list` command after using the `task stop` command to display information about the task, as in the following example:

## *task Commands*

```
AAC0>task list  
Executing: task list  
No tasks currently running on controller.
```

## **Related Information**

task commands:

- task list ([page 9-2](#))

## task suspend

To suspend one or all tasks running on the currently opened controller, use the `task suspend` command.

### Syntax

```
task suspend [/all{=boolean}]
```

### Parameters

{integer}

Specifies the task ID for the task to suspend. Perform a `task list /all` to display the task IDs for all tasks currently running on the open controller before suspending a specific task or all tasks.

### Switches

/all{=boolean}

Specifies whether to suspend all running tasks. If you set this switch to `TRUE`, the command suspends all tasks that were running on the currently opened controller.

If you do not specify this switch or a valid task ID, the command displays an appropriate error message.

### Examples

Before suspending a running task, use the `task list` command to identify any currently running tasks, as in the following example:

```
AAC0>task list

Executing: task list
TaskId  Function  Done% Array      State  Specific1  Specific2
-----  -
101     Scrub      24.6%  0             RUN    00000000  00000000
```

The following example shows the use of the `task suspend` command to suspend task 101:

```
AAC0> task suspend 101

Executing: task suspend 101
```

Use the `task list` command after using the `task suspend` command to display information about the task, as in the following example:

```
AAC0>task list
```

```
Executing: task list
```

TaskId	Function	Done%	Array	State	Specific1	Specific2
101	Scrub	24.6%	0	SUS	00000000	00000000

As a result of using the `task suspend` command, the `State` column now displays `SUS` (suspended on the controller) instead of `RUN` (running on the controller).

## Related Information

task commands:

- `task list` ([page 9-2](#))
- `task resume` ([page 9-8](#))



## Automated Scripts

This appendix describes how to write DOS batch files and CLI command scripts to perform the `controller details` task.

You can use the CLI commands alone in CLI command scripts or in combination with DOS commands in DOS batch files. Also, you can invoke CLI command scripts in a DOS batch file. You should already understand how to write DOS batch files and be familiar with Windows-related backup commands.

To execute a CLI command script, type the @ (at sign) followed by the file name that contains the CLI commands, for example:

```
AAC0>@array.txt
```

### Creating an Automated Script

In its simplest form, a CLI command script is a text file that contains valid CLI commands. A carriage return linefeed follows each command. The following CLI command script:

- 1 Opens controller `aac0` by invoking the CLI command `open`.
- 2 Invokes `controller details`.
- 3 Closes controller `aac0` by invoking the CLI command `close`.
- 4 Exits the example CLI command script.

```
open AAC0
controller details
close AAC0
exit
```

*Automated Scripts*





**Adaptec, Inc.**  
691 South Milpitas Boulevard  
Milpitas, CA 95035 USA

©2000 – 2003 Adaptec  
All rights reserved. Adaptec and the Adaptec logo  
are trademarks of Adaptec which may be  
registered in some jurisdictions.

Part Number: 5138123-06, Ver. AA 11/03