AMCC STORAGE

3ware®

Serial ATA RAID Controller

Command Line Interface

Supports the 9000 Series

PN: 720-0104-00
April, 2004

CLI Guide

Copyright

Trademarks

Disclaimer

Table of Contents

About This Guide

3ware 9000 Series Serial ATA Controller CLI Guide provides instructions for configuring and maintaining your 3ware controller using 3ware's command line interface (CLI).

This guide assumes that you have already installed your controller in your system. If you have not yet done so, see *3ware 9000 Series Serial ATA RAID Controller Installation Guide* for instructions.

How this Guide is Organized

There are often multiple ways to accomplish the same configuration and maintenance tasks for your 3ware controller. While this manual includes instructions for performing tasks using the command line interface, two additional tools are available:

3ware BIOS Manager

3DM®2 (3ware Disk Manager)

For information about these tools, see *3ware 9000 Series Serial ATA RAID Controller User Guide*.

Table 1: Sections in this Guide

Section	Description
Introduction to 3ware Command Line Interface	Installation, Features, Concepts
Reference	Describes Individual Commands

Conventions

The following conventions are used through this guide:

- 3BM refers to the 3ware BIOS Manager
- 3DM refers to the 3ware Disk Manager, version 2.
- `Monospace font` is used for code and for things you type.
- In commands, an italic font indicates items that you must specify, such as a controller ID, or a unit ID.
- In commands, brackets around an item indicates that it is optional.
- In commands, ellipses (. . .) indicate that more than one parameter can be included.
- In commands, a brace (|) indicates an 'or' situation where the user has a choice between more than one option, but only one can be specified.

Additional details are provided in “CLI Reference” on page 15.

Introduction to the 3ware Command Line Interface

The 3ware SATA RAID Controller Command Line Interface (CLI) for Linux, Windows, and FreeBSD is provided to manage 7000, 8000, and 9000-series 3ware ATA and Serial ATA RAID controllers. Multiple 3ware controllers can be managed using the CLI via a command line or script. CLI is useful in environments where a graphical user interface (GUI) is not available.

Note: All information contained in this document that describes usage for the 3ware 9000 series products should not be used with 3ware 7000 or 8000 series controllers.

Warning!

For all of the functions of the 3ware CLI to work properly, you must have the proper CLI, firmware, and driver versions installed. Check www.3ware.com for the latest versions and upgrade instructions.

Features

3ware CLI is a command line interface storage management application for 3ware ATA RAID Controllers. It provides controller, logical unit, and drive management. It can be used in both interactive and batch mode, providing higher level API functionalities.

The 3ware CLI provides the functionality of the 3ware Disk Management (3DM®2) utility through a command line interface. You can use it to view unit status and version information and perform maintenance functions such

as adding or removing drives. 3ware CLI also includes advanced features for creating and deleting RAID units online.

Note: For complete information on 3DM 2 and for information about configuring or upgrading your computer, refer to the *3ware 9000 Series Serial ATA RAID Controller User Guide*.

Supported Operating Systems

- **Windows.** Windows 2000 with SP3 or newer, Windows XP with SP1 or newer, and Windows Server 2003.
- **Linux**. Redhat, SuSE
- **FreeBSD**

Note: The 3ware CLI support for FreeBSD is currently a beta release.

Terminology

This document uses the following terminology:

Logical Units. Usually shortened to "units." These are block devices presented to operating systems. A logical unit can be a one-tier, two-tier, or three-tier arrangement. JBOD, Spare, and Single logical units are examples of one-tier units. RAID 1 and RAID 5 are examples of two-tier units and as such will have sub-units. RAID 10 and RAID 50 are examples of three-tier units and as such will have sub-sub-units.

Port. A controller has one or many ports (typically 4, 8, 12). Each port can be attached to a single disk drive.

For additional information about 3ware controller concepts and terminology, see *3ware 9000 Series Serial ATA RAID Controller User Guide*.

Installing the 3ware CLI

Warning!

3ware does not recommend installing both 3DM 2 and CLI. Conficts may ocur. For example, if both are installed, alarms will be captured only by 3DM. You should use either CLI or 3DM 2 to manage your 3ware RAID controllers..

For all of the functions of the 3ware CLI to work properly, you must have the proper CLI, firmware, and driver versions installed. Check www.3ware.com for the latest versions and upgrade instructions.

Installing 3ware CLI on Windows

3ware CLI can be run from the 3ware CD, or copied to your computer from the 3ware software CD-ROM. You can also download the CLI from the 3ware web site, www.3ware.com.

To install 3ware CLI on Windows, copy the file `tw_cli.exe` to the directory from which you want to run the program. CLI is located on the 3ware CD in the directory `\packages\cli\windows`

Note: CLI comes in both 32-bit and 64-bit versions. Be sure to copy the correct version.

CLI can only be run by an administrator or a user with administrator rights. Without the correct privileges, CLI will prompt and then exit when the application is executed.

To start CLI, do one of the following:

- Start the 3ware CD and at the 3ware Escalade menu, click **Run CLI.**
- Or, open a console window and at the command prompt, enter `tw_cli`
- OR, double-click the CLI icon in a folder.

  The CLI prompt is displayed in a DOS console window.

Installing the 3ware CLI on Linux and FreeBSD

3ware CLI can be installed from the 3ware software CD-ROM, or downloaded from the 3ware web site, www.3ware.com.

You will need to be root or have root privileges to install the CLI to `/usr/sbin` and to run the CLI.

```
Filename: tw_cli.tar
```

To install the CLI, type the following as root:

```
tar xf tw_cli.tar -C /usr/sbin
```

To install the CLI to a different location, change `/usr/sbin/` to the desired location.

Note: The installation location needs to be in the environment path for root to execute the CLI without using complete paths (i.e., if installed to `/usr/sbin/`, you can type **`tw_cli`** on the command line, otherwise you will have to type the complete path: **`/home/user/tw_cli`**

Working with 3ware CLI

You can work with the 3ware CLI in different ways:

- Interactively, entering commands at the main prompt
- As a series of single commands
- By creating a script—an input file with multiple commands

This first section shows examples of each of these ways.

Examples shown in the *CLI Reference* chapter reflect the interactive method.

Using the command interface interactively

You can use 3ware CLI interactively, entering commands at the main prompt and observing the results on the screen.

To use the CLI interactively

1 Enter the following command:

```
# tw_cli
```

The main prompt is displayed, indicating that the program is awaiting a command.

```
3ware CLI>
```

2 At the CLI prompt, you can enter commands to get, set, or maintain 3ware controllers.

For example

```
3ware CLI>info
```

Displays all of the controllers in the system.

To display details of a single controller (assumed to be `c0)` and its attached units, enter:

```
3ware CLI>info c0
```

To display details of a single unit (assumed to be `u0)` on the single controller, enter:

```
3ware CLI>info c0 u0
```

Using a single command with output

You can use 3ware CLI with line arguments, processing a single command at a time. To do so, simply enter the command and the arguments.

Syntax

```
tw_cli <command line arguments>
```

Example

```
tw_cli info c0 u0
```

This example gets information for unit0 of controller0. For complete information about the info commands, see “Info Commands” on page 18.

Using an input file to execute a script

You can operate 3ware CLI scripts by executing a file. The file is a text file containing a list of CLI commands which you have entered in advance. Each command must be on a separate line.

Syntax

```
tw_cli -f <filename>
```

Where `<filename>` is the name of the text file you want to execute.

Example

```
tw_cli -f clicommand.txt
```

This example executes the file `clicommand.txt`, and runs the CLI commands included in that file.

Scripting Example

Here is an example using a text file called `info.txt`, containing the following two commands:

```
info c0 u0
info c1 u0
```

To run the script, enter:

```
tw_cli -f info.txt
```

3ware CLI displays information about unit 0 on controller 0 and unit 0 on controller 1.

Understanding RAID Concepts and Levels

The next few pages introduce RAID concepts you may find useful. For additional information about installing and managing your 3ware controller, see the *3ware 9000 Series Serial ATA RAID Controller User Guide*.

3ware controllers use a Redundant Array of Inexpensive Disks (RAID) to increase your storage system’s performance and provide fault tolerance (protection against data loss).

The following concepts are important to understand when working with a RAID controller:

- **Arrays and Units**. In the storage industry, the term "array" is used to describe two or more disk drives that appear to the operating system as a single unit. When you work with 3ware software, "unit" is the term used to refer to an array of disks that is configured and managed through the 3ware software. Single-disk units can also be configured in the 3ware software.
- **Mirroring**. Mirrored arrays write data to paired drives simultaneously. If one drive fails, the data is preserved on the paired drive. Mirroring provides data protection through redundancy. In addition, mirroring using a 3ware controller provides improved performance because 3ware's TwinStor technology reads from both drives simultaneously.
- **Striping**. Striping across disks allows data to be written and accessed on more than one drive, at the same time. Striping concatenates each drive's capacity into one large volume. Striped disk arrays achieve high transfer rates and provide improved performance.
- **Distributed Parity**. Parity works in combination with striping on RAID 5 and RAID 50. Parity information is written to each of the striped drives, in rotation. Should a failure occur, the data on the failed drive can be reconstructed from the data on the other drives.
- **Hot Swap**. The process of swapping out a drive without having to shut down the system. This is useful when you need to swap out a degraded drive, manually or automatically, with a pre-designated spare.
- **Array Roaming.** The process of swapping out or swapping in a configured unit without having to shut down the system. This is useful if you need to move the unit to another controller.
- **Disk Roaming.** The process of removing a unit from a controller and putting it back later, either on the same controller, or a different one, and having it recognized as a unit. The disks may be in a different order than they initially occupied, without harm to the data.

Available RAID Configurations

The following RAID levels and configurations are available for drives attached to a 3ware controller:

- RAID 0
- RAID 1
- RAID 5
- RAID 10
- RAID 50
- Single Disk
- JBOD
- Hot Spare

RAID 0

Provides striping, but no mirroring. Striped disk arrays achieve high transfer rates because they can read and write data on more than one drive simultaneously. The stripe size is configurable in the 3ware CLI, 3ware BIOS Manager (3BM) and in the 3ware Disk Manager (3DM 2). Requires a minimum of two drives.

When drives are configured in a striped disk array (see Figure 1), large files are distributed across the multiple disks using RAID 0 techniques.

Striped disk arrays give exceptional performance, particularly for data intensive applications such as video editing, computer aided design and geographical information systems.

RAID 0 arrays are not fault tolerant. The loss of any drive results in the loss of all the data in that array.

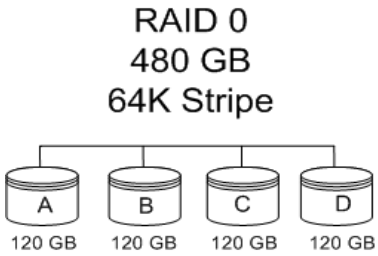

Figure 1. RAID 0 Configuration Example

RAID 1

Also known as a mirrored array. Mirroring is done on pairs of drives. Mirrored disk arrays write data to two drives using RAID 1 algorithms (see Figure 2). This gives your system fault tolerance by preserving the data on one drive if the other drive fails. Fault tolerance is a basic requirement for mission critical systems like web and database servers.

3ware uses a patented technology, TwinStor®, on RAID 1 arrays for improved performance during sequential read operations. With TwinStor technology, read performance is twice the speed of a single drive during sequential read operation.

The adaptive algorithms in TwinStor technology boost performance by distinguishing between random and sequential read requests. For the sequential requests generated when accessing large files, both drives are used, with the heads simultaneously reading alternating sections of the file. For the smaller random transactions, the data is read from a single optimal drive head.

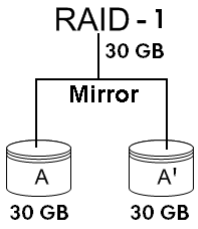

Figure 2. RAID 1 Configuration Example

RAID 5

Combines striping data with parity (exclusive OR) to restore data in case of a drive failure. This array type provides performance, fault tolerance, high capacity, and storage efficiency. Requires a minimum of three drives.

Parity information is distributed across all drives rather than being concentrated on a single disk (see Figure 3). This avoids throughput loss due to contention for the parity drive.

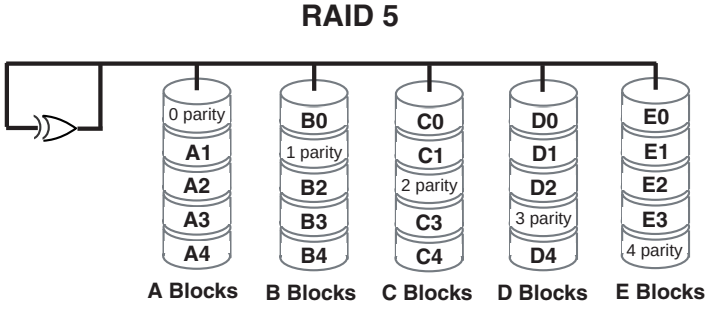

Figure 3. RAID 5 Configuration Example

RAID 10

This array is a combination of RAID 1 with RAID 0. Striped and mirrored arrays for fault tolerance and high performance. Requires a minimum of four drives to use both RAID 0 and RAID 1 techniques.

When drives are configured as a striped mirrored array, the disks are configured using both RAID 0 and RAID 1 techniques, thus the name RAID 10 (see Figure 4). A minimum of four drives are required to use this technique. The first two drives are mirrored as a fault tolerant array using RAID 1. The third and fourth drives are mirrored as a second fault tolerant array using RAID 1. The two mirrored arrays are then grouped as a striped RAID 0 array using a two tier structure. Higher data transfer rates are achieved by leveraging TwinStor and striping the arrays. RAID 10 is available on the four, eight, and twelve port 3ware Serial ATA RAID Controllers.

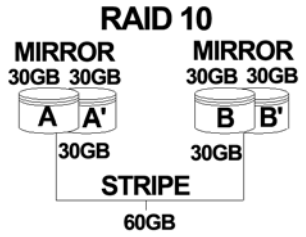

Figure 4. RAID 10 Configuration Example

RAID 50

This array is a combination of RAID 5 with RAID 0. This array type provides fault tolerance and high performance. Requires a minimum of six drives.

Several combinations are available with RAID 50. For example, on a 12-port controller, you can have a grouping of 3, 4, or 6 drives. A grouping of 3 means that the RAID 5 arrays used have 3 disks each; four of these 3-drive RAID 5 arrays are striped together to form the 12-drive RAID 50 array.

Single Disk

A single drive that has been configured as a unit through 3ware software. Like disks in other RAID configurations, single disks contain 3ware Disk Control Block (DCB) information and are seen by the OS as available units.

JBOD

A JBOD is an unconfigured disk attached to your 3ware RAID controller. JBOD configuration is no longer supported in the 3ware 9000 series. 3ware recommends that you use Single Disk as a replacement for JBOD, to take advantage of advanced features such as caching. If you are migrating JBODS from a 7000/8000 controller, you can enable support for them. For more information, contact Technical Support

Hot Spare

A single configured drive, available so that a redundant array can be rebuilt in case of drive failure.

For additional information about RAID levels, see the article “RAID Primer” on the 3ware website, at: http://www.3ware.com/products/pdf/RAID_Primer.pdf.

Determining What RAID Level to Use

Select the RAID configuration to use based on the applications to be used on the system, whether performance or data protection is of primary importance, and the number of disk drives available for use.

Review the information under "Understanding RAID Concepts and Levels" on page 7 to determine the type of RAID configuration most appropriate for your needs and use the tables below to determine what RAID levels are available, based on your particular controller model and the number of available drives.

The RAID configurations available to you are determined by the number of ports on your controller, and the number drives attached to those ports. You can configure all drives in one unit, or you can configure multiple units, if you have enough drives.

Table 2: Possible Configurations Based on Number of Drives

# Drives	Possible RAID Configurations
1	Single or spare drive
2	RAID 0 or RAID 1
3	RAID 0, RAID 5, or RAID 1 + spare
4	RAID 5 + hot spare RAID 10 Combination of RAID 0, RAID 1, single disk
5	RAID 5 + hot spare RAID 10 + hot spare Combination of RAID 0, RAID 1, hot spare
6 or more	RAID 50 Depending on the number of drives, a RAID 50 may contain from 2 to 4 subunits. For example, with 12 drives, possible RAID 50 configurations include 2 subunits of 6, 3 subunits of 4, or 4 subunits of 3. With 10 drives, a RAID 50 will contain 2 subunits of 5 drives each. Combination of RAID 0, 1, 5, 10, hot spare, and single disk

Drive Capacity Considerations

The capacity of each drive is limited to the capacity of the smallest drive in the array. The total array capacity is defined as follows:

Table 3: Drive Capacity

RAID Level	Capacity
RAID 0	(number of drives) X (capacity of the smallest drive)
RAID 1	capacity of the smallest drive
RAID 5	(number of drives - 1) X (capacity of the smallest drive) Storage efficiency increases with the number of disks: storage efficiency = (number of drives -1) / (number of drives)
RAID 10	(number of drives / 2) X (capacity of smallest drive)
RAID 50	(number of drives - number of groups of drives) X (capacity of the smallest drive)

CLI Reference

This chapter provides detailed information about the 3ware CLI commands Info, Maint, Sched, Alarms, Set, and Help.

Note: All information contained in this document that describes usage for the 3ware 9000 series products should not be used with 3ware 7000 or 8000 series controllers.

Info. Information commands provide all information and settings about the 3ware controllers, including array types, array status, array settings, detail controller information, and detail drive information. For details, see "Info Commands" on page 18.

Maint. Maintenance commands perform all maintenance operations on the drives and arrays connect to the 3ware controller. Typical operations include: create array, delete array, rebuild array, verify array, and remove array from the controller. For details, see "Maint Commands" on page 28.

Warning: Operations under the maint command can destroy data, so care should be taken before using this command; CLI does not prompt before the operation is committed.

Sched. Scheduling commands only affect the 9000 controller. Schedule commands allow you to schedule different time slots for background tasks such as rebuild, verify, and selftest. For details, see "Sched Commands" on page 36.

In order to use CLI scheduling commands with 7000 and 8000 controllers, you must use them in conjunction with a time-driven scheduling component under Windows, Linux, or FreeBSD. For example, under Linux, you can use the cron daemon scheduling utility with the CLI commands for rebuild, verify, and mediascan. (For more information about the specific CLI commands, see

"Maint Commands" on page 28 and refer to your Linux documentation or manpages.)

In addition, 3ware also includes on the 3ware CD a utility named tw_sched(1), which is a wrapper around tw_cli(1). Used in conjunction with a time-driven scheduler such as crond(1d), it provides background task scheduling features such as rebuild, verify, and mediascan. For details about tw_sched(1), see the manpages for it.

Scheduling for the 7000 and 8000 series models can also be done using 3DM 1.x. (Note that 3DM 2 only provides scheduling only for 9000 series models.)

Alarms. The Alarms command allows you to display Asynchronous Event Notification (AEN) events that have been generated by controllers. AEN events have different levels of severity. They can be extracted and archived for overall trend analysis. For details, see "Alarms Commands" on page 41.

CLI does not store alarms command in a log file, so when you reboot, alarms in the previous session will be lost. To preserve the alarms through reboot, you can either extract the alarms from CLI and store them in a file, or install 3DM 2, which does log alarm messages. For more information, see the *3ware 9000 Series Serial ATA RAID Controller User Guide*.

Set. Setting commands can be used to modify and change controller and array settings. Settings that can be changed include: rebuild rate, verify rate, and turning on or off cache, autoverify, or overwriteECC. For details, see "Set Commands" on page 43.

Help. Help commands. Options in this category allow you to display help information on the other commands and options. For details, see "Help Commands" on page 45.

Throughout this chapter the examples reflect the interactive method of executing 3ware CLI.

Conventions

The following conventions are used through this guide:

- In text, `monospace font` is used for code and for things you type.
- In commands, an italic font indicates items that you must specify, such as a controller ID, or a unit ID.
- In commands, brackets around an item indicates that it is optional.
- In commands, ellipses (. . .) indicate that more than one parameter can be included.
- In commands, a brace (|) indicates an 'or' situation where the user has a choice between more than one option, but only one can be specified.

For example, in the maint command to rescan all ports and reconstitute all units, the syntax appears as `maint rescan [cid ...] [noscan]`. The italic *cid* indicates that you need to supply a controller ID. The ellipses indicate that you can specify more than one controller ID, separated by spaces. The brackets indicate that you may omit the controller ID, to rescan all controllers, and the noscan parameter, so that the operation will be reported to the operating system.

Screen Reporting Style

Beginning with this version of CLI, 3ware has changed the default reporting style to a tabular reporting style for screen displays. Using this format, information is easier to read and analyze. The new style also accommodates automation, by providing consistent columns with or without values so that it can be easily parsed.

The original, non-tabular style is still available for a limited time. To use the old style, set the TW_CLI_STYLE to OLD as shown below, depending on your operating system.

- For Redhat and SuSE (bash, ksh, or sh), enter
  `export TW_CLI_STYLE=OLD`
- For Linux csh (C-shell), enter:
  `setenv TW_CLI_STYLE OLD`
- For Windows, enter
  `set TW_CLI_STYLE=OLD`

To keep the new CLI output style following a reboot or when a new window or shell is opened you must edit the environment variables in both Windows and Linux.

To use the new style, enter
`TW_CLI_STYLE=" "`

or
`TW_CLI_STYLE="NEW"`

The examples in this document use the new style of reporting.

Info Commands

The *info* commands provide information about the 3ware controller, the attached drives, and configured RAID arrays or units. The *info* commands are for querying purposes only.

Info commands are read-only operations showing various values of controllers, units, and drives.

Syntax

```
info
info c<c> [driver|model|firmware|bios|monitor|serial|pcb|pchip|achip
    numports|numunits|numdrives|unitstatus|drivestatus|allunitstatus
    exportjbod|ondegrade|spinup|stagger]
  info c<c> u<u> [status|rebuildstatus|verifystatus|initializestatus]
  info c<c> p<p> [status|model|serial|capacity|smart]
  info c<c> diag
```

Parameters

cid - the controller id

uid - the unit id

pid - the port id

option - specifies the kind of information you want to see.

info

Provides information on all detected controllers. The appropriate device driver must be loaded for the list to show all controllers. The intention is to provide a global view of the environment.

Typical output looks like the following:

```
3ware CLI> info
Ctl    Model      Ports   Drives  Units  NotOpt    RRate   VRate
------------------------------------------------------------------
c0     7500-12    12      5       1      1         2       -
c1     8506-12    12      6       1      0         3       5
```

The output indicates that controller 0 is a 7000 series with 12 ports, 5 drives, and a total of 1 unit in a not optimal state. Not optimal refers to any state except OK and VERIFYING. Other states include INITIALIZATING, REBUILDING, DEGRADED, MIGRATING, and INOPERABLE. The example shows that the controller's rebuild rate (RRate) is set to 2 and the verify rate (VRate) is not applicable (-).

Additional attributes about individual controllers, units, ports and disks can be obtained by querying for them explicitly, using, for example, `info cid` or `info cid uid`. See the other *info* sub-commands below.

info *cid*

Provides overall summary information on controller *cid*. The report consists of two parts; a unit summary listing all present units, and a port summary section listing all present disks and their attached ports.

The unit summary section lists all present units specifying their unit number, unit type (such as RAID 5), status, size (usable capacity) in gigabytes or terabytes, number of blocks, and unit status such as OK, VERIFYING, INITIALIZING, etc. %Compl reports percent completion of REBUILDING or VERIFYING units. It shows the Stripe size, if applicable, and whether Cache is on or off. It also shows whether AutoVerify and OvrECC are on or off (9000 only). OvrECC is short for Overwrite ECC, or Force Continue on source error. The function is explained in "Set Commands" on page 43.

The port summary section lists all present ports specifying the port number, disk status, unit affiliation, size (GB), blocks (of 512 bytes), and the serial number assigned by the disk vendor.

Additional attributes about units, ports and disks can be obtained by querying for them explicitly. See other info sub-commands below.

Typical output looks like:

```
3ware CLI> info c0
Unit  UnitType Status %Cmpl  Stripe Size   Cache AVerify  OvrECC
------------------------------------------------------------------
u0    RAID-1   OK     -      -      149.05 ON    OFF      OFF
u1    RAID-5   OK     -      64k    298.22 ON    OFF      OFF
u2    SPARE    OK     -      -      149.05 ON    OFF      -

Port    Status        Unit  Size         Blocks       Serial
---------------------------------------------------------------
p0      OK            u0    149.05 GB    312581808    3JS0TF14
p1      OK            u0    149.05 GB    312581808    3JS0TETZ
p2      OK            u1    149.05 GB    312581808    3JS0VG85
p3      OK            u1    149.05 GB    312581808    3JS0VGCY
p4      OK            u1    149.05 GB    312581808    3JS0VGGQ
p5      OK            u2    149.05 GB    312581808    3JS0VH1P
p6      OK            -     149.05 GB    312581808    3JS0TF0P
p7      OK            -     149.05 GB    312581808    3JS0VF43
p8      OK            -     149.05 GB    312581808    3JS0VG8D
p9      NOT-PRESENT   -     -            -            -
p10     NOT-PRESENT   -     -            -            -
p11     NOT-PRESENT   -     -            -            -
```

3ware CLI calculates one megabyte as 1024 x 1024, the same calculation that Windows and Linux use. 3DM 2 uses 1024 x 1024, so when using 3DM 2, the capacity listed will match the capacity stated by the CLI. Previous versions of 3DM (v1.x) calculate one megabyte as 1000 x 1000, which is the calculation disk drive vendors use.

info *cid driver*

This command reports the device driver version associated with controller *cid*.

Example:

```
3ware CLI> info c0 driver
/c0 Driver Version = 1.02.00.036
```

info *cid model*

This command reports the controller model of controller *cid*.

Example:

```
3ware CLI> info c0 model
/c0 Model = 7506-12
```

info *cid firmware*

This command reports the firmware version of controller *cid*.

Example:

```
3ware CLI> info c0 firmware
/c0 Firmware Version = FGXX 2.01.00.025
```

info *cid bios*

This command reports the BIOS version of controller *cid*.

Example:

```
3ware CLI> info c0 bios
/c0 BIOS Version = BG9X 2.01.00.026
```

info *cid monitor*

This command reports the monitor (firmware boot-loader) version of controller *cid*.

Example:

```
3ware CLI> info c0 monitor
/c0 Monitor Version = BLDR 1.00.00.008
```

info *cid serial*

This command reports the serial number of the specified controller *cid*.

Example:

```
3ware CLI> info c0 serial
/c0 Serial Number = F12705A3240009
```

info *cid pcb*

This command reports the PCB (printed circuit board) revision of the specified controller *cid*.

Example:

```
3ware CLI> info c0 pcb
/c0 PCB Version = Rev3
```

info *cid pchip*

This command reports the PCHIP (PCI Interface Chip) version of the specified controller *cid*.

Example:

```
3ware CLI> info c0 pchip
/c0 PCHIP Version = 1.30-33
```

info *cid achip*

This command reports the ACHIP (ATA Interface Chip) version of the specified controller *cid*.

Example:

```
3ware CLI> info c0 achip
/c0 ACHIP Version = 3.20
```

info *cid numports*

This command reports the number of ports of the specified controller *cid*.

Example:

```
3ware CLI> info c0 numports
/c0 Number of Ports = 12
```

info *cid numunits*

This command reports the number of units currently managed by the specified controller *cid*. This report does not include units that have been removed (placed off-line) with the maint remove command.

Example:

```
3ware CLI> info c0 numunits
/c0 Number of Units = 1
```

info *cid numdrives*

This command reports the number of drives currently managed by the specified controller *cid*. This report does not include units that have been removed (placed off-line) with the maint remove command.

Also note that a physically removed disk is not detected unless I/O is performed against the disk. See "info cid pid smart" on page 26 for a workaround.

Example:

```
3ware CLI> info c0 numdrives
/c0 Number of Drives = 5
```

info *cid unitstatus*

This command presents status of units managed by the specified controller *cid*. It provides a list of units, their types, current status, percent complete if rebuilding or verifying, size in GB, and the number of blocks of 512 bytes.

Example:

```
3ware CLI> info c0 unitstatus
Unit  UnitType Status %Cmpl  Stripe Size(GB)Cache AVerify OvrECC
-------------------------------------------------------------
u0    RAID-5 VERIFYING   79    16K  819.446   ON    OFF    ON
```

info *cid allunitstatus*

This command presents a count of Total and NotOptimal units managed by the specified controller *cid*. See "Info Commands" on page 18 for more information on NotOptimal.

Example:

```
3ware CLI> info c0 allunitstatus
Total Units = 2
NotOptimal Units = 0
```

info *cid drivestatus*

This command presents a list of port assignments, status, unit affiliation, size in GB, the number of blocks of 512 bytes, and the disk's serial number.

Example:

```
3ware CLI> info c0 drivestatus
Port    Status        Unit  Size          Blocks       Serial
---------------------------------------------------------------
p0      OK            u0    149.05 GB     312581808    3JS0TF14
p1      OK            u0    149.05 GB     312581808    3JS0TETZ
p2      OK            u1    149.05 GB     312581808    3JS0VG85
p3      OK            u1    149.05 GB     312581808    3JS0VGCY
p4      OK            u1    149.05 GB     312581808    3JS0VGGQ
p5      OK            u2    149.05 GB     312581808    3JS0VH1P
p6      OK            -     149.05 GB     312581808    3JS0TF0P
p7      OK            -     149.05 GB     312581808    3JS0VF43
p8      OK            -     149.05 GB     312581808    3JS0VG8D
p9      NOT-PRESENT   -     -             -            -
p10     NOT-PRESENT   -     -             -            -
p11     NOT-PRESENT   -     -             -            -
```

info *cid exportjbod*

This command shows whether the Export JBOD policy is enabled or disabled.

Example:

```
3ware CLI> info c0 exportjbod
/c0 JBOD Export Policy = off
```

info *cid ondegrade*

This command shows whether the write cache will be disabled if a unit degrades.

Example:

```
3ware CLI> info c0 ondegrade
/c0 Cache on Degrade Policy = Follow Unit Policy
```

info *cid spinup*

This command shows whether staggered spinup is enabled.

Example:

```
3ware CLI> info c0 spinup
/c0 Disk Spinup Policy = 1
```

Note that "1" indicates enabled. When disabled, "255" is shown.

info *cid stagger*

This command shows the delay between drive groups that spin up at one time on this controller

Example:

```
3ware CLI> info c0 stagger
/c0 Spinup Stagger Time Policy (sec) = 2
```

info *cid uid*

This command presents detailed information on the specified unit. If the unit consists of sub-units as is the case in RAID 1, RAID 5, RAID 10, and RAID 50 arrays (applicable for 9000 controllers), then details about each sub-unit are also presented. One application of this command is to see which sub-unit of a degraded unit has caused the unit to degrade and which disk within that sub-unit is the source of degradation.

Example:

```
3ware CLI> info c0 u0

Unit   UnitType  Status  %Cmpl  Port  Stripe  Size(GB)  Blocks
-----------------------------------------------------------------
u0     RAID-5    VERIFYING  0     -     16K   819.446 718503424
u0-0   DISK      OK         -     p0    -     74.4951 156227584
u0-1   DISK      OK         -     p1    -     74.4951 156227584
u0-2   DISK      OK         -     p2    -     74.4951 156227584
u0-3   DISK      OK         -     p3    -     74.4951 156227584
u0-4   DISK      OK         -     p4    -     74.4951 156227584
u0-5   DISK      OK         -     p5    -     74.4951 156227584
u0-6   DISK      OK         -     p6    -     74.4951 156227584
u0-7   DISK      OK         -     p7    -     74.4951 156227584
u0-8   DISK      OK         -     p8    -     74.4951 156227584
u0-9   DISK      OK         -     p9    -     74.4951 156227584
u0-10  DISK      OK         -     p10   -     74.4951 156227584
u0-11  DISK      OK         -     p11   -     74.4951 156227584
```

info *cid uid status*

This command presents the status of the specified unit.

Example:

```
3ware CLI> info c0 u5 status
/c0/u5 status=DEGRADED
```

info *cid uid rebuildstatus*

This command presents the rebuild status (if any) of the specified unit.

Example:

```
3ware CLI> info c0 u5 rebuildstatus
/c0/u5 is not rebuilding.
```

Or, when the unit is rebuilding:

```
3ware CLI> info c0 u5 rebuildstatus
/c0/u5 is rebuilding with Percent Completion = %14
```

info *cid uid verifystatus*

This command presents the verify status (if any) of the specified unit.

Example:

```
3ware CLI> info c0 u5 verifystatus
/c0/u5 is not verifying.
```

info *cid uid initializestatus*

This command presents the initialize status (if any) of the specified unit.

Example:

```
3ware CLI> info c0 u5 initializestatus
/c0/u5 is not initializing.
```

info *cid pid*

This command presents various information on the specified disk attached to port *pid*. Typical information looks like:

Example:

```
3ware CLI> info p5
Port   Status    Unit     Size        Blocks        Serial
---------------------------------------------------------------
p5     OK        u2       149.05 GB   312581808     3JS0VH1P
```

This report indicates that port 5 of controller 0 is attached to disk serial number 3JS0VH1P, with status OK participating in unit 5.

info *cid pid status*

This command presents the status of the specified port.

Example:

```
3ware CLI> info c0 p5 status
/c0/p5 Status = OK
```

info *cid pid model*

This command presents the model of the specified port.

Example:

```
3ware CLI> info c0 p5 model
/c0/p5 Model = WDC WD1600BB-00DAA0
```

info *cid pid serial*

This command presents the serial number of the specified port.

Example:

```
3ware CLI> info c0 p5 serial
/c0/p5 Serial = WD-WMACK1406498
```

info *cid pid capacity*

This command presents the capacity, both in human readable form (such as GB) and block count of the specified port. Note that capacity is computed based on division by 1024 (not 1000 as is popular with hard disk vendors). For additional information, see the explanation at “info cid” on page 19.

Example:

```
3ware CLI> info c0 p5 capacity
/c0/p5 Capacity = 149.05 GB (312581808)
```

info *cid pid smart*

This command extracts SMART (Self Monitoring Analysis and Reporting) data from the specified disk. The data is extracted live from the disk; therefore, this command is used to get the most recent data about the presence or absence of a disk. The SMART data is displayed in hexadecimal form. Since SMART data is extracted live from this disk, it places a burden on the I/O bandwidth.

Example:

```
3ware CLI> info c0 p5 smart
10 00 01 0B 00 C8 C8 00 00 00 00 00 00 00 03 07
00 9A 96 BC 14 00 00 00 00 00 04 32 00 64 64 7A
00 00 00 00 00 00 05 33 00 C8 C8 00 00 00 00 00
...
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2C
```

info *cid diag*

This command extracts the internal log of diagnostic events of controller *cid,* controller diagnostics suitable for technical support usage. Note that some characters might not be printable or rendered correctly (human readable). It is recommended to save this output to a file, where it can be communicated to tech support.

Example:

```
 3ware CLI> info c0 diag

 --Port[ 5]-
 DIT status: DRV_PRESENT (0xFF)
  Model #: Maxtor 6Y080M0
  Serial #: Y3LLQWPE
  Drv FW #: YAR51EW0
  Capacity: 160086528 (0x098ABA00)
  Features: SMART: 1, Security: 1, 48-bit addr: 0, Acoustic: 1,
 Feat. Ext: TimeLimited R/W: 0, WDMA FUA: 0, Stream: 0
  Acoustic: 0xFE, def=0xC0 (0xFE=best performance)
  Security: Status=0x7 (ENABLED, LOCKED)
  SATA NCQ: 0
 Udma Mode: 0x5 (UDMA-100)
Pwr Cycles: 14
….
SELF TEST: all tests completed.
UA::SpareUnit(0) checking for spare needed
UA::SpareUnit done.
Auto Clean: 0
DcbMgr::UpdateStatus: UNIT  0 (time   7704019)
DcbMgr::WriteSegment(map=0xFFE, segID=0x8, events=22,
error=0x0)
DcbMgr::UpdateStatus:         (finish 7704024)
SETTINGS: Saving USER parameters ... done. (count=1, bytes=9)
SETTINGS: Saving USER parameters ... done. (count=1, bytes=9)
SETTINGS: Saving USER parameters ... done. (count=1, bytes=9)
SETTINGS: Saving USER parameters ... done. (count=1, bytes=9)
Send AEN (code, time): 0x31, 0x7a2d6b
Synchronize host/controller time
(EC:0x31, SK=0x00, ASC=0x00, ASCQ=0x00, SEV=04, Type=0x71)
```

Maint Commands

The *maint* command lets you perform maintenance operations on the controller, its units, and drives. It is recommended that you use the *info* command first to verify the controller information before using the *maint* command to make any changes to it.

Sub-commands under this category allow you to create and modify objects and their attributes such as creating and deleting logical units, rebuilding, etc. These commands are read/write operations and should be used with care.

Use of the keyword "maint" is now optional. For example, "`maint rescan c0`" is the same as "`rescan c0`".

Syntax

```
[maint]         (Note: maint keyword is now optional)
  rescan [c<c> ...] [noscan] |NOTE: Does not import non-
    JBOD on 7000/8000 models.
  remove c<c> u<u> [noscan]
  remove c<c> p<p>
  deleteunit c<c> u<u> [noscan]
  createunit c<c> r<RaidType> p<p:-p..> [k<stripe>] [nos-
    can] [nocache] [g<3|4|5|6>] [autoverify] [ignoreECC]
    RaidType = { raid0, raid1, raid5, raid10, raid50, sin-
    gle, spare }
  rebuild c<c> u<u> p<p:-p...> [ignoreECC]
  rebuild c<c> u<u> pause  (7000/8000 only)
  rebuild c<c> u<u> resume  (7000/8000 only)
  flush c<c> [u<u>]
  verify c<c> u<u> [stop]
 mediascan c<c> start|stop (7000/8000 only)
 commit c<c> (** Windows only **)
```

[maint] *rescan* [cid ...] [noscan]

This command instructs the controller to rescan all ports, and reconstitute all units. The controller updates its list of ports (attached disks), and visits every Disk Configuration Block (DCB) in order to re-assemble its view and awareness of logical units.

If no controller is specified, all controllers are rescanned. One or several controllers can be specified.

By default, the OS is informed of changes resulting from rescan. You can alter this behavior using the noscan option.

Rescan imports JBOD units only when attached to either a 7000 or 8000 controller, unless you reboot. All other RAID types can be imported when attached to the 9000 series.

Warning!

Adding any drive requires use of an approved hot swap carrier. If you do not have such a carrier you must first power down your system. Failure to do so may cause the system to hang or become corrupted. It may even damage your system.

Example:

```
3ware CLI> maint rescan
Rescanning controller /c0 for units and drives ...Done.
Rescanning controller /c1 for units and drives ...Done.
```

If you use the noscan option:

```
3ware CLI>maint rescan c0 noscan
```

Using the noscan option allows a system administrator to export units to the OS a later time rather than having the CLI do it for them.

[maint] remove *cid uid* [noscan]

This command allows you to remove (or export) a unit. Exporting a unit instructs the firmware to remove the specified unit from its poll of managed units, but retains the Disk Configuration Block (DCB) metadata.You can import (re-introduce) the unit via rescan. By default the OS is informed of this change. You can alter this behavior using the noscan option.

Warning!

You must first unmount the array before issuing the maint remove command. Failure to do so may cause the system to hang or become corrupted.

Warning!

Physically removing any drive requires use of an approved hot swap carrier. If you do not have such a carrier you must first power down your system. Failure to do so may cause the system to hang or become corrupted. It may even damage your system.

[maint] remove *cid pid* [noscan]

This command allows you to remove (or export) a port (or drive). Exporting a port instructs the firmware to remove the specified port from its poll of managed ports, but retains the Disk Configuration Block (DCB) metadata on the attached disk. You can import (re-introduce) the port via rescan. By default the OS is informed of this change. If you use the noscan option the OS is not notified of the drive removal.

Warning!

Removing any drive requires use of an approved hot swap carrier. If you do not have such a carrier you must first power down your system. Failure to do so may cause the system to hang or become corrupted. It may even damage your system.

Alert!

Removing a drive causes a redundant array to degrade. Drives cannot be removed if they are part of a degraded or non-redundant array, with the exception of Single and JBOD drives.

Warning! Single and JBOD Drives

You must first unmount any Single or JBOD drive before issuing the remove command. Failure to do so may cause the system to hang or become corrupted.

[maint] deleteunit *cid uid* [noscan]

This command allows you to delete a unit. Deleting a unit not only removes the specified unit from the controller's list of managed units, but also destroys the DCB (Disk Configuration Block) metadata. Ports (or disks) associated with this unit will now be part of the free poll of managed disks. This is a destructive command and should be used with care. By default the OS is informed of this change. You can alter this behavior using the noscan option.

Warning! Back up data

Back up any critical data prior to deleting a unit. Failure to do so will result in lost data.

[maint] createunit *cid rRAIDType pid_list* [kStripe] [noscan] [Dsk_Grp] [nocache] [autoverify] [ignoreECC]

This command allows you to create a unit on the specified controller *cid*, of type *rRAIDType*, optional stripe size of *kStripe*, using one or many disks specified by *pid_list*. By default the host operating system is informed of the new block device and write cache is enabled. In case of RAID 50, you can also specify the layout of the unit by specifying the number of disks per disk group with the *gDsk_Grp* option.

cid is the controller name as in c0, c1, etc.

rRAIDType is the RAID or Logical Unit type as in RAID 0, RAID 1, RAID 5, RAID 10, RAID 50, single, spare, and JBOD. The following table illustrates supported types and controller models.

Table 4: Supported RAID Types

Model	R-0	R-1	R-5	R-10	R-50	Single	JBOD	Spare
7K/8K	Yes	Yes	Yes	Yes	N/A	N/A	Yes	Yes
9K	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

pid_list is a list of ports (disks) to be used in the construction of the specified unit. One or more ports can be specified. Multiple ports can be specified using a colon (:) to separate port indexes and a dash (-) to include a range of port indexes. A dash indicates a range and can be mixed with colons. For example p0:1:2-5:9:12 indicates port 0, 1, 2 through 5 (inclusive), 9 and 12.

kstripe indicates the stripe size to be used. The following table illustrates the supported and applicability of stripes on unit types and controller models. Stripe size units are in K (kilobytes).

Table 5: Supported Stripe Sizes

Model	R0	R1	R5	R10	JBOD	Spare	R50	Single
7K/8K	64	N/A	64	64	N/A	N/A	N/S	N/S
	128			128				
	256			256				
	512			512				
	1024			1024				
9K	16	N/A	16	16	N/A	N/A	16	N/A
	64		64	64			64	
	256		256	256			256	

gdsk_grp indicates the number of disks per group for a RAID 50 type. A RAID 50 is a multi-tier array. At the most bottom layer, N number of disks per group are used to form the RAID 5 layer. These RAID 5 arrays are then integrated into a RAID 0. This option allows you to specify the number of disks in the RAID 5 level. Valid values are 3, 4, 5 and 6.

Note that a sufficient number of disks are required for a given pattern or disk group. For example, given 6 disks, specifying 3 creates two RAID 5 arrays. However given 12 disks, specifying 3 creates four RAID 5 arrays under the RAID 0 level. Given 6 disks, specifying 6 is not allowed as you'll basically be creating a RAID 5.

The default RAID 50 grouping **(gdsk_grp)** varies based on number of disks. For 6 and 9 disks, the default grouping is 3. For 8 disks the default grouping is 4. For 10 disks the default grouping is 5 and for 12 disks, the default grouping is 4. In the case of 12 disks could be grouped into groups of 3, 4 or 6 drives. A grouping of 4 was set by default as it provides best of net capacity and performance.

noscan switch instructs CLI not to notify OS of the creation of the new unit. By default CLI informs the OS. One application of this feature is to avoid OS creating block special devices such as /dev/sdb, /dev/sdc as some implementations might create naming fragmentation and creating a moving target.

nocache switch instructs CLI not to enable the write cache. Enabling write cache increases performance at the cost of high-availability.

autoverify switch enables the autoverify attribute on the unit that is to be created. This feature is not supported on model 7000/8000. Autoverify is used in conjunction with the scheduling option. If autoverify is enabled, the array is verified repeatedly during a scheduled verify window. If autoverify is disabled, verify is not initiated by the controller and must be started manually.

ignoreECC switch enables the ignoreECC/OverwriteECC attribute on the unit that is to be created. The following table illustrates the supported Model-UnitType. This table only applies to setting this feature at Unit Creation time. Generally ignoreECC applies to redundant units.

Table 6: Supported Model-Unit Types

Model	R-0	R-1	R-5	R-10	R-50	Single	JBOD	Spare
7K/8K	No	No	No	No	No	No	No	No
9K	No	Yes	Yes	Yes	No	No	Yes	No

Examples:

To create a 12-member RAID-0 array with 128K stripe size on controller 0:

```
CLI> maint createunit c0 rraid0 k256 p0-11
```

To create a hot spare using a drive on port-2 controller-0 for automatic rebuilds:

```
CLI> maint createunit c0 rspare p2
```

Alert!

When creating a hot spare, be sure to select a drive with an equal or larger size than the smallest drive in your redundant array. Otherwise it can't be used in a rebuild.

[maint] rebuild *cid uid pid_list* [ignoreECC]

This command allows you to rebuild a DEGRADED unit by using the specified port. Rebuild applies only to redundant arrays such as RAID 1, RAID 5, RAID 10 and RAID 50.

During rebuild, bad sectors on the source disk cause the rebuild to fail. Using the ignoreECC option (equivalent to checking the 'Force continue on source errors' box in 3DM) allows the rebuild to continue when source errors occur, but a file system check is recommended once the rebuild is complete.

Note: The ignoreECC option is not required for the 9000 series, if the variable is already assigned when you create or set later. Refer to the "set overwriteECC cid uid on|off" on page 44 for more info.

The rebuild process is a background process and changes the state of a unit to REBUILDING. Various *info* commands also show a percent completion as rebuilding progresses.

Ports that are to be used to rebuild a unit must be a Spare type or an unconfigured drive. You must first use `remove` to remove the failed drive, then use `rescan` to add the new drive before you can use `rebuild`.

[maint] rebuild *cid uid* pause

This command allows you to pause the rebuild operation on the specified unit. This feature is intended for model 7000 and 8000 only. Model 9000 has an on-board scheduler where rebuild operations can be scheduled to take place at specified start and stop times. Rebuild pause and resume function is provided to enable 7000/8000 users to achieve similar functionality with the use of OS-provided schedulers.

See also "Sched Commands" on page 36.

[maint] rebuild *cid uid* resume

This command allows you to resume the rebuild operation on the specified unit. See "[maint] rebuild cid uid pause" on page 33 for more details.

[maint] flush *cid* [*uid* ...]

This command allows you to flush the write cache on the specified unit or all units associated with controller *cid*. This command does not apply to Spare unit types.

[maint] verify *cid uid* [stop]

This command starts or stops a background verification process on the specified unit. To start, omit "stop".

The following table shows the supported RAID types for verification as a function of controller model and logical unit type. N/A (Not Applicable) refers to cases where the given logical unit type is not supported on a particular controller model.

Table 7: Supported RAID Types for Verification

Model	R-0	R-1	R-5	R-10	R-50	Single	JBOD	Spare
7K/8K	No	Yes	Yes	Yes	N/A	N/A	No	No
9K	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

[maint] mediascan *cid* start|stop

This command applies to 7000/8000 controllers. It provides media scrubbing for validating the operationality of a disk, including bad block detection and remapping. The start or stop operations start or stop media scan operation on the specified controller. For the 9000 series, the verify function includes the features of the media scan command.

[maint] commit *cid*

This command only applies to the Windows operating system.

This command instructs the controller to commit its dirty DCBs to persistent storage (disks). While the controller is processing I/O requests against underlying disks, an in-transaction bit is set. If a failure (such as power failure) is experienced, subsequent reads from the disk inform the controller that an un-clean shutdown took place. This command allows the end user to complete all pending I/Os on disks and clear the in-transaction bit.

Typical application of this feature is when an application is using a given unit in raw mode (such as databases) and the user would like to shut down the host (including UPS post failure automations). This command expedites the process by instructing the controller to finish pending requests and clear the DCBs in-transaction flag as the disk is going down.

Note that block devices (cooked devices) do not require this command. Clients of block devices (such as File system) send such requests via ASPI SRB_SHUTDOWN_REQUEST.

Sched Commands

Sched commands are applicable for 9000-series controllers to schedule background tasks to occur at a later time or day, when peak performance is not required. When the schedule is disabled, by default, background task occur almost immediately. Background tasks include rebuild, verify, and selftest activities. For each activity, up to 7 tasks can be registered, known as slots 0 through 6. Each task can be managed with these commands including adding, removing, enabling and disabling a task. Background tasks have slot id, category (rebuild, verify, selftest), start time, and duration attributes.

rebuild activity is the process of rebuilding one or more DEGRADED units to one or more specified ports. Rebuild applies only to redundant arrays such as RAID 1, RAID 5, RAID 10 and RAID 50. Initialization activity of new arrays is also included with this background task.

verify activity visits every unit of a given controller and attempts to verify all members of redundant units. On the 9000 series, non-redundant units, including spares, are also verified by doing a background scrub which reads each sector. Verifying RAID 1 involves checking that both drives contain the exact data. On RAID 5, the parity information is used for error correction. RAID 10 and 50 are composite types and follow their respective array types.

selftest activity provides two types of selftests: Ultra Direct Memory Access (UDMA) and Self Monitoring Analysis and Reporting (SMART). UDMA selftest checks the current ATA bus speed (between the controller and an attached disk) which could have been throttled down during previous operations and increases the speed for best performance (usually one level higher). Possible speeds include 33, 66, 100 and 133 MB/s.

SMART activity instructs the controller to check certain SMART-supported thresholds by the disk vendor. The UDMA selftest is not required for serial ATA drives.

Syntax

```
sched rebuild c<c>
sched rebuild c<c> add d<d> h<h> t<t>
sched rebuild c<c> remove <n>
sched rebuild c<c> enable|disable
sched verify c<c>
sched verify c<c> add d<d> h<h> t<t>
sched verify c<c> remove <n>
sched verify c<c> enable|disable
sched selftest c<c>
sched selftest c<c> add d<d> h<h>
sched selftest c<c> remove <n>
sched selftest c<c> enable|disable s<s>
```

sched rebuild *cid*

This command displays the current rebuild background tasks as illustrated below.

```
3ware CLI> tw_cli sched rebuild c1
Rebuild Schedule for controller /c1
=======================================================
Slot    Day     Hour            Duration        Status
-------------------------------------------------------
0       Mon     2:00pm          10 hr(s)        disabled
1       Thu     7:00pm          18 hr(s)        disabled
2       -       -               -               -
3       -       -               -               -
4       -       -               -               -
5       Mon     1:00am           4 hr(s)        disabled
6       Sun     12:00am          1 hr(s)        disabled
```

sched rebuild *cid* add *day hour duration*

This command adds a new background rebuild task to be executed on day (range 0 .. 6, where Sunday is zeroth day of the week), at hour (range 0 .. 23), for a duration of duration (range 1 .. 24) hours. This command will fail if no (empty) slot is available.

Note: The new schedule is added to the first available slot. Events do not need to be added in sequential order

For example:

```
3ware CLI> tw_cli sched rebuild c1 add d0 h16 t3
```

Adds a rebuild background task to be executed on Sundays at 4:00 PM for a duration of 3 hours.

sched rebuild *cid* remove slot_id

This command removes (or unregisters) the rebuild background task in slot slot_id.

Warning: If all timeslots are removed, be sure to also disable the schedule. Otherwise the applicable background task will never occur.

For example:

```
3ware CLI> tw_cli sched rebuild c1 remove 2
```

Removes the rebuild background task in slot 2.

sched rebuild *cid* enable

This command enables ALL rebuild background tasks on controller *cid*.

sched rebuild *cid* disable

This command disables ALL rebuild background tasks on controller *cid*.

sched verify *cid*

This command displays the current verify background task as illustrated below.

```
3ware CLI> tw_cli sched verify c1
Verify Schedule for controller /c1
=======================================================
Slot    Day     Hour            Duration        Status
-------------------------------------------------------
0       Mon     2:00am          4 hr(s)         disabled
1       -       -               -               -
2       Tue     12:00am         24 hr(s)        disabled
3       Wed     12:00am         24 hr(s)        disabled
4       Thu     12:00am         24 hr(s)        disabled
5       Fri     12:00am         24 hr(s)        disabled
6       Sat     12:00am         24 hr(s)        disabled
```

sched verify *cid* add day hour duration

This command adds a new background verify task to be executed on day (range 0 .. 6, where Sunday is zeroth day of the week), at hour (range 0 .. 23), for a duration of duration (range 1 .. 24) hours. This command will fail if no (empty) slot is available.

For exmaple:

```
3ware CLI> tw_cli sched verify c1 add d0 h16 t3
```

Adds a verify background task to be executed on Sundays at 4:00 PM for a duration of 3 hours.

Note: The new schedule is added to the first available slot. Events do not need to be added in sequential order.

sched verify *cid* remove *slot_id*

This command removes (or unregisters) the verify background task in slot *slot_id*.

For example:

```
3ware CLI> tw_cli sched verify c1 remove 3
```

Removes the verify background task in slot 3.

Warning: If all timeslots are removed, be sure to also disable the schedule. Otherwise the applicable background task will never occur.

sched verify *cid* enable

This command enables all verify background tasks on controller *cid*.

Note: When enabling the verify schedule you must also remember to also enable the autoverify setting for the arrays to be verified.

sched verify *cid* disable

This command disables all verify background tasks on controller *cid*.

sched selftest *cid*

This command displays the current selftest background task as illustrated below.

```
3ware CLI> tw_cli sched selftest c1
Selftest Schedule for controller /c1
=======================================================
Slot    Day     Hour            UDMA            SMART
-------------------------------------------------------
0       Sun     12:00am         enabled         enabled
1       Mon     12:00am         enabled         enabled
2       Tue     12:00am         enabled         enabled
3       Wed     12:00am         enabled         enabled
4       Thu     12:00am         enabled         enabled
5       Fri     12:00am         enabled         enabled
6       Sat     12:00am         enabled         enabled
```

sched selftest *cid* add *day hour*

This command adds a new background selftest task to be executed on day (range 0 .. 6, where Sunday is zeroth day of the week), at hour (range 0 .. 23). Notice that selftest runs to completion and as such no duration is provided. This command fails if no (empty) slot is available.

For example:

```
$ tw_cli sched selftest c1 add d0 h16
```

Adds a selftest background task to be executed on Sundays at 4:00 PM.

Note: The new schedule is added to the first available slot. Events do not need to be added in sequential order. Also the selftests are completed almost

sched selftest *cid* remove *slot_id*

This command removes (or unregisters) the selftest background task in slot *slot_id.*

For example:

```
3ware CLI> tw_cli sched selftest c1 remove 3
```

Removes rebuild selftest task in slot 3.

Warning: If all timeslots are removed, be sure to also disable the schedule. Otherwise the applicable background task will never occur

sched selftest *cid* enable *selftest_task_id*

This command enables a particular selftest_task (UDMA or SMART). Selftest_task_id s0 is interpreted as UDMA; Selftest_task_id s1 is interpreted as SMART.

Note: When enabling the verify schedule you must also remember to also enable the autoverify setting for the arrays to be verified.

For example:

```
3ware CLI> tw_cli sched selftest c1 enable s0
```

Enables UDMA selftest on controller c0.

sched selftest *cid* disable *selftest_task_id*

This command disables a particular selftest task (UDMA or SMART). For the selftest_task_id, s0 is interpreted as UDMA, s1 is interpreted as SMART.

For example:

```
3ware CLI> tw_cli sched selftest c1 disable s1
```

Disables SMART selftest on controller c1.

Alarms Commands

The *alarms* command provides a log of alarms, also called Asynchronous Event Notifications (AENs), that have occurred on the disk arrays. An alarm occurs when the ATA RAID controller requires attention, such as when a disk array becomes degraded and is no longer fault tolerant. SMART notifications appear in this display. Alarm messages are categorized by the following levels of severity:

- Errors
- Warnings
- Information

When the *alarms* command is executed, only AENs that have been logged since the last time the command was executed are displayed. For Linux, AENs are also saved in a text file at `/var/log/messages`.

Windows users can see the AEN messages in the Windows System Event Logs that can be seen in the Event Viewer.

Asynchronous events are originated by firmware and captured by their respective device drivers. These events are kept in a finite queue inside the kernel, awaiting extraction by user programs such as CLI or 3DM 2. These events reflect warning, debugging, or informative messages for the end user.

Alarms generated on 7000/8000 models do not have dates, so a dash (-) meaning *not-applicable* appears in the Date column. Also on 7000/8000 models, the alarm message does not contain the severity, hence the Severity column displays a dash (-) as well.

Syntax

```
alarms
alarms c<c> [c<c> ...]
```

Warning!

3ware does not recommend installing both 3DM and CLI. Conflicts may occur. If both are installed, alarms will be captured only by 3DM.

Warning!

3DM and CLI handle alarms differently. When using CLI with the 7/8000 series, save the alarm data immediately after viewing it. Once the alarms are viewed using CLI, they cannot be viewed again.

With the 9000 series, the alarms can be viewed multiple times with the CLI, but will be lost following a reboot. At this time there is no CLI option to clear the alarms log.

alarms [*cid* ...]

This command displays all available alarms on a single controller, multiple controllers, or on all controllers. Invoked without *cid*, displays alarms associated with all detected controllers.

Note: A listing of AEN codes can be found in the "Troubleshooting: Problems and Solutions" section of *3ware 9000 Series Serial ATA RAID Controller User Guide*.

Typical output looks like:

```
tw_cli> alarms

Ctl  Date                  Severity  Message
--------------------------------------------------
c0   -              -         ERROR: Unit degraded: Unit #0
c1 [Fri Nov 28 04:26:31 2003] ERROR (0x04:0x0002): Degraded unit detected: unit=0, port=2
c1 [Fri Nov 28 06:13:54 2003] INFO  (0x04:0x000B): Rebuild started: unit=0
c1 [Fri Nov 28 06:30:35 2003] INFO  (0x04:0x003B): Background rebuild paused: unit=0
c1 [Fri Nov 28 06:33:00 2003] ERROR (0x04:0x0002): Degraded unit detected: unit=0, port=0
c1 [Fri Nov 28 06:33:04 2003] ERROR (0x04:0x0002): Degraded unit detected: unit=0, port=4
c1 [Fri Nov 28 06:33:46 2003] INFO  (0x04:0x000B): Rebuild started: unit=0
c1 [Fri Nov 28 06:37:58 2003] INFO  (0x04:0x000B): Rebuild started: unit=0
c1 [Fri Nov 28 07:51:34 2003] INFO  (0x04:0x0005): Background rebuild done: unit=0
c1 [Fri Nov 28 07:59:43 2003] INFO  (0x04:0x0005): Background rebuild done: unit=0
c1 [Mon Dec  1 02:26:12 2003] ERROR (0x04:0x0002): Degraded unit detected: unit=0, port=3
```

Set Commands

These commands allow you to set certain controller and unit specific parameters as described below. The set command can be used to set its rebuild rate, and enable or disable cache. For information about viewing information about the controller and units, see the “Info Commands” on page 18.

Syntax

```
set rebuild c<c> <1..5>
set cache c<c> u<u> on|off
set verify  c<c> <1..5>                (Note: 9000 only)
set autoverify c<c> u<u> on|off        (Note: 9000 only)
set overwriteECC c<c> u<u> on|off      (Note: 9000 only)
```

Parameters

- *rebuild* - sets the rebuild rate (per controller basis) and also sets the initialize rate.
- *cache* - enables or disables caching on a per array or unit basis for RAID 1, 5, and 10 arrays.
- *verify* - sets the verify rate (per controller basis)
- *autoverify* - enables or disables the automatic verification (per unit basis)
- *overwriteECC* - enables or disables the ignoreECC function dring rebuild (per unit basis)

Note: A value of 1 indicates slowest I/O and fastest rebuild rate. A value of 5 indicates fastest I/O and slowest rebuild. Interim values scale linearly (e.g., a value of 3 indicates a rebuild rate half as fast as a rebuild of 1).

set rebuild *cid 1..5*

This command allows you to set the priority of rebuild in relation to I/O operations. Setting this value to 1 implies that rebuilds should consume more resources (cpu time, I/O bandwidth) to complete its task. Conversely, setting this value to 5 implies that I/O has higher priority and rebuild. This command applies to 7000, 8000, and 9000 models. For the 7/8000 series, the rebuild rate also applies to the verify and background scrub tasks.

set verify *cid 1..5*

This command allows you to set the priority of verification in relation to I/O operations. Setting this value to 1 implies fastest verify, and 5 implies fastest I/O. Note that this feature only applies to 9000 models.

set cache *cid uid on|off*

This command allows you to turn on or off the write cache on a specified unit. This feature is supported on both 7000/8000 and 9000 models.

The following table shows the supported RAID types for caching as a function of controller model and logical unit type. N/A (Not Applicable) refers to cases where the given logical unit type is not supported on a particular controller model.

Table 8: Supported RAID Types for Caching

Model	R-0	R-1	R-5	R-10	R-50	Single	JBOD	Spare
7K/8K	Yes	Yes	Yes	Yes	N/A	N/A	Yes	No
9K	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No

set autoverify *cid uid on|off*

This command allows you to turn on or off the verify operation on a specified unit during times specified by sched verify commands. The sched verify command allows you to specify times for the verify operation, it does not associate the operation with a unit. This command allows you to associate a unit with the verify operation. This feature only applies to 9000 models.

set overwriteECC *cid uid on|off*

Setting overwriteECC to on means ignoreEcc. This command allows you to set the ignoreECC policy for a given unit. This policy is then automatically enforced during a rebuild of the specified unit. This option can also be specified when rebuilding the array manually as discussed in the maintenance section. If this setting is already enabled you do not need to specify it again when rebuilding an array. This setting only applies to 9000 models.

Help Commands

This command set provides brief on-line help.

help

This command provides a table of contents, providing brief descriptions of the help sub-commands. Typical output looks like:

```
 Copyright (c) 2003 3ware, Inc.  All rights reserved.
 List of Commands
 ----------------------------------------
 info   - displays information about the controller
 alarms - displays or deletes the list of AENs
 set    - displays or modifies controller settings
 maint  - performs maintenance operations on a controller
 sched  - Sets the schedule for a controller (9000 controllers
only)
 quit   - exits the CLI
 Type help <command> to get more details about a particular
command.
```

help info

This command provides specific *info* related help, illustrating various ways to use the info command. Info provides reports on 3ware controllers, units and drives.

help alarms

This command provides specific *alarms* related help, illustrating various ways to use the alarms command.

help set

This command provides specific *set* related help, illustrating various ways to use the set command.

help maint

This command provides specific *maint* related help, illustrating various ways to use the maint command.

help sched

This command provides specific *sched* related help, illustrating various ways to use the sched command. Applies to the 9000 version only.

help quit

This command provides information about the CLI *quit* command. For example:

```
3ware CLI> help quit
This command quits the CLI

  quit

    Synonyms: q exit
```

Return Code

While informative messages are written to standard output, error messages are written to standard error. On success, 0 is returned. On failure, 1 is returned.

To view the return code, at the shell command prompt type:

```
echo $?
```

The screen prints either a 0 or a 1, depending on whether the command was successful or not.

For example, if you had a 3ware controller with an ID of 0, you could type this command:

```
tw_cli info c0
(c0 information displayed here)
echo $?
0
```

If you type:

```
tw_cli info c7
error: (CLI003) specified controller does not exist.
echo $?
1
```

This example fails (returns 1) because there is no controller 7.