Solaris™ Reference Manual for SMCC-Specific Software

Solaris 2.6 Hardware: 3/98
Contents

Preface v

1. User Commands (1) 1-1
   smc_copy 1-1
   smc_ls 1-3
   smc_rm 1-4
   smc_tar 1-5
   symon 1-7

2. Maintenance Commands (1M) 1M-10
   afbconfig 1M-10
   cfgadm 1M-18
   cvcd 1M-27
   dr_daemon 1M-28
   luxadm 1M-29
   sm_configd 1M-40
   sm_confsymon 1M-41
   sm_control 1M-43
   sm_egd 1M-44
   sm_krd 1M-46
   sm_logscand 1M-47
sm_symond 1M-48
smc_ftpd 1M-50
smc_gettacl 1M-54
smc_settacl 1M-55
ssp-config 1M-57
ssp-unconfig 1M-58
sunvts 1M-59
vtsk 1M-60
vtsprobe 1M-61
vtstty 1M-64
vtsui 1M-66
vtsui.ol 1M-67

3. Miscellaneous Library Functions (3) 3-68
   config admin (3X), 3-68

4. File Formats (4) 4-75
   auth_checker.tcl 4-75
   auth_list.tcl 4-76
   event_gen.tcl 4-77
   logscan.tcl 4-78
   rules.tcl 4-79
   sm_symond.conf 4-80

5. Devices and Network Interfaces (7) 7-81
   afb 7-81
   cvc 7-82
   cvcredir 7-83
   dr 7-84
   idn 7-85
Preface

The Solaris Reference Manual for SMCC-Specific Software contains manual pages (man pages) for software provided to SMCC customers with the Solaris 2.6 Hardware: 3/98 product. These supplement the man pages provided in the general Solaris 2.6 Reference Manual.

Before you can access some of the information published in this book through the man command, you may need to install software from the SMCC Supplement CD for your Solaris release. In most cases, when you install a software cluster from the SMCC Supplement CD, man pages about the software in that cluster will be automatically installed. For information about installing the man page software, refer to the Vendor Value-Added Software section of the Solaris Information Library for your Solaris release.

How This Book Is Organized

This manual contains manual pages in alphabetical order within each man page category. Supplemental man pages are included for the following categories:
- User Commands (1)
- Maintenance Commands (1M)
- Miscellaneous Library Functions (3)
- File Formats (4)
- Device and Network Interfaces (7)
Ordering Sun Documents

SunDocs™ is a distribution program for Sun Microsystems technical documentation. Contact SunExpress for easy ordering and quick delivery. You can find a listing of available Sun documentation on the World Wide Web.

### TABLE P-1  SunExpress Contact Information

<table>
<thead>
<tr>
<th>Country</th>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belgium</td>
<td>02-720-09-09</td>
<td>02-725-88-50</td>
</tr>
<tr>
<td>Canada</td>
<td>1-800-873-7869</td>
<td>1-800-944-0661</td>
</tr>
<tr>
<td>France</td>
<td>0800-90-61-57</td>
<td>0800-90-61-58</td>
</tr>
<tr>
<td>Germany</td>
<td>01-30-81-61-91</td>
<td>01-30-81-61-92</td>
</tr>
<tr>
<td>Holland</td>
<td>06-022-34-45</td>
<td>06-022-34-46</td>
</tr>
<tr>
<td>Japan</td>
<td>0120-33-9096</td>
<td>0120-33-9097</td>
</tr>
<tr>
<td>Luxembourg</td>
<td>32-2-720-09-09</td>
<td>32-2-725-88-50</td>
</tr>
<tr>
<td>Sweden</td>
<td>020-79-57-26</td>
<td>020-79-57-27</td>
</tr>
<tr>
<td>Switzerland</td>
<td>0800-55-19-26</td>
<td>0800-55-19-27</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>0800-89-88-88</td>
<td>0800-89-88-87</td>
</tr>
<tr>
<td>United States</td>
<td>1-800-873-7869</td>
<td>1-800-944-0661</td>
</tr>
</tbody>
</table>


Sun Documentation on the Web

The docs.sun.com web site enables you to access Sun technical documentation on the World Wide Web. You can browse the docs.sun.com archive or search for a specific book title or subject at [http://docs.sun.com](http://docs.sun.com).
Sun Welcomes Your Comments

We are interested in improving our documentation and welcome your comments and suggestions. You can email your comments to us at smcc-docs@sun.com. Please include the part number of your document in the subject line of your email.
NAME

smc_copy – copies content from one Sun MediaCenter server to another

SYNOPSIS

smc_copy [-p] [-s] [-t] source destination

where source and destination each have the form hostname:filename> or
<filename>. Specifying only <filename> implies that a title is stored on or being
copied to the machine from which you are invoking smc_copy. Options are
described below.

AVAILABILITY

Available with the Sun MediaCenter Server software.

DESCRIPTION

The smc_copy utility copies content, specified by a title name as returned by smc_is
(1), from one Sun MediaCenter server to another.

The syntax for smc_copy is similar to that of rcp (1), with the following exceptions:

-- You cannot specify a path to a title, in either the source or destination argument. Sun MediaCenter software looks for and stores titles and associated data
in hardcoded locations.

-- You cannot specify a username in an smc_copy source or destination argument.

As with rcp, you can invoke smc_copy on one machine to copy content from a second
machine to a third machine. The machine on which you invoke smc_copy does not
have to be a Sun MediaCenter server. To run smc_copy, you must have the
smc_copy binary installed on the local machine.

In the course of the copy operation, smc_copy creates a new title on a destination Sun
MediaCenter server. You can rename the destination title in an smc_copy command.

When you copy a title to another Sun MediaCenter, you, the copier, own the title
access control list for that file. Other users can play the title, but cannot copy it to
another server or remove it. To extend access to the newly-copied file, use smc_settacl
(1M).

OPTIONS

The smc_copy utility has the following options:

p  Preserve create time in destination file. Otherwise, the create time becomes the
current time for the newly copied file.

s  Run in single-threaded mode. By default, the utility runs in multiple threads.
This option is used for internal test purposes.

Display transfer statistics in shell from which you invoke the utility.

EXAMPLES

The following example copies the title heidi from the Sun MediaCenter server server2
to the local Sun MediaCenter server, server1.

    server1% smc_copy server2:heidi heidi
The following command accomplishes the same function as the preceding:

```
server1% smc_copy server2:heidi server1:heidi
```

The following command copies content from a local to a remote Sun MediaCenter server, renaming the title in the process:

```
server1% smc_copy heidi server2:drama
```

The following command performs the same function as the preceding, except the title is not renamed:

```
server1% smc_copy heidi server2:heidi
```

The following command copies the title *heidi* from Sun MediaCenter server *server1* to the Sun MediaCenter server *server2* and renames the title in the process. The command is invoked from a third-party machine, *machine_x*, which is not an Sun MediaCenter server.

```
machine_x% smc_copy server1:heidi server2:drama
```

SEE ALSO

```
smc_tar (1), smc_ls (1), smc_rm (1), smc_setacl (1M), smc_getacl (1M), smc_ftpd (1M)
```
NAME        smc_ls – list playable titles on a Sun MediaCenter server

SYNOPSIS    smc_ls [smc_svr_name]

AVAILABILITY Available with the Sun MediaCenter Server software.

DESCRIPTION The smc_ls list the titles available for playback on a local or remote Sun MediaCenter server. You can play these titles through the facilities of the Media Stream Manager. For each title stored on a Sun MediaCenter server, smc_ls returns the title name, the normal play time, the available playback speeds, and an indication of whether the title is in use or is free.

OPTIONS    The smc_ls command allows you to specify the name of a remote Sun MediaCenter server, to obtain a title list from that server. To run smc_ls remotely, you need only the smc_ls binary, which is installed with the Sun MediaCenter software.

EXAMPLES    The following example lists all playable content on a local Sun MediaCenter server:

```
server% smc_ls

Title     Status  NPT         Format      Available Speeds
terminator2 cm 01:52:30  MPEGTCE  1000,4000,-4000
dr_zhivago FREE 02:48:21  MPEG1SYS 1000,4000,-4000
mary_poppins cm,msm 02:03:17  MPEGPS  1000,-1000
```

Note, under "Available Speeds", that "1000" represents normal speed, forward direction. A value "4000" represents four times normal speed; "-4000" represents four times normal speed in the reverse direction.

Under "Status", FREE indicates the title is not in use. The string "cm" indicates the title is in use by the Content Manager (for example, if it is being copied to another server). The string "msm" indicates the title is being played (by the Media Stream Manager).

SEE ALSO    smc_tar (1), smc_rm (1), smc_ftpd (1M)
NAME  smc_rm – remove content from Media File System on a Sun MediaCenter server

SYNOPSIS  smc_rm [smc_svr_name:]<title1> [smc_svr_name:]<title2> ...

AVAILABILITY  Available with the Sun MediaCenter Server software.

DESCRIPTION  The smc_rm removes content from the Media File System (MFS) on a Sun MediaCenter server. The command takes as an argument one or more titles. Optionally, each title can be prepended with the name of a remote Sun MediaCenter server. smc_rm removes a specified title, including the index file and all MPEG files referred to by that title.

You can run smc_rm on a remote machine that is not a Sun MediaCenter server. All that is required to run the utility is the smc_rm binary, which you can copy from a Sun MediaCenter server.

OPTIONS  The smc_rm command allows you to specify a remote Sun MediaCenter server for each title specified in a command line.

EXAMPLES  The following example removes all content associated with the titles Bambi, on the local Sun MediaCenter server, and Quo Vadis, on the Sun MediaCenter server named "vidserver".

server% smc_rm bambi vidserver:quo_vadis

SEE ALSO  smc_tar (1), smc_ls (1), smc_copy (1)
NAME
smc_tar – move content between tar device or file and the Media File System on Sun MediaCenter server

SYNOPSIS
smc_tar t|c|x[v][b][w] f device [blksize]

AVAILABILITY
Available with the Sun MediaCenter Server software.

DESCRIPTION
The smc_tar command loads properly prepared multimedia content from a tar device, such as an 8mm tape, or a file onto the Media Filesystem (MFS). Content must prepared according to the rules specified in the Sun MediaCenter software documentation. These rules include the following:

- a single title per tar device;
- a Table of Contents (TOC) file for each title;
- an index file for each title;
- a separate MPEG stream for each playback speed and direction different from normal speed, forward direction.

In the course of loading content, smc_tar parses the TOC file, does error-checking with respect to the index file, and converts the MPEG bit streams to MFS files.

Note that ftp, in conjunction with the Sun MediaCenter ftp daemon, is the preferred method of loading content onto a Sun MediaCenter server.

With the c option, you can use smc_tar to back up content from a Sun MediaCenter server to tar device or file.

You can use smc_tar from a machine that is not a Sun MediaCenter server and from a remote Sun MediaCenter server, to move content between a server and a local or remote tar device or file. You need only the smc_tar binary, available on a Sun MediaCenter server, to run the utility.

The smc_tar command is analogous to the Unix filesystem tar (1) utility.

smc_tar has a single mandatory argument, f, which precedes the name of the tar device.

OPTIONS

b blksize
Where blksize is the block size that was used to create the tar contents. blksize must be a multiple of 20 and, if present, is the last argument in the smc_tar command line. One block equals 512 bytes. The recommended block size is 500, which is 256000 bytes. Most operating systems, including Solaris, have a default block size of 20.

c Specifies creation of a tar file or copying a title from the Sun MediaCenter server to a tar device. Requires a source file argument, one or more of <server>:<title>, where <title> can be the wildcard asterisk, meaning all titles on a server. Used primarily for backup.

t Display a table of contents of the specified tar device or file.

modified 15 April 1997

Solaris 2.6 Hardware: 3/98

1-5
v  Verbose. Display progress of command.
w  Prompt user before overwriting already-existing content.
x  Specifies extraction from the named tar device or file.
<device>
   The tar device from which you are extracting content.

**EXAMPLES**
The following example loads content from the tar device `/dev/rmt/0`, specifying a block size of 40 and prompting you before overwriting existing files:

```
server% smc_tar xwb 40 /dev/rmt/0
```

The following command copies all of the files on a remote server to a local tape device:

```
remote_host% smc_tar cf /dev/rmt/0 smc_server:
```

Note that you must use a backslash (`\`) to escape the asterisk.

The following command gives you a table of contents for the titles stored in a tape device on a remote Sun MediaCenter server:

```
host% smc_tar tfv remote_server:/dev/rmt/0
```

**SEE ALSO**
tar (1), smc_ls (1), smc_rm (1), smc_settacl (1M), smc_gettacl (1M), smc_ftpd (1M)
NAME  symon – bring up the Solstice SyMON system monitor console

SYNOPSIS  symon [ -colorMap ] [ -cm ] [ *colorMap ]
           [ -dragthreshold pixels ] [ *dragthreshold pixels ]
           [ -flashDuration milliseconds ]
           [ -fd milliseconds ] [ *flashDuration milliseconds ]
           [ -flashInterval milliseconds ] [ -fi milliseconds ] [ *flashInterval milliseconds ]
           [ -heartbeatInterval intervals ] [ -hi intervals ] [ *heartbeatInterval intervals ]
           [ -interval intervals ] [ -i intervals ] [ *interval intervals ]
           [ -installDir path ] [ -I path ] [ *installDir path ]
           [ -minWait seconds ] [ -mw seconds ] [ *minWait seconds ]
           [ -pruneTime minutes ] [ -pt minutes ] [ *pruneTime minutes ]
           [ -session file ] [ *session file ] [ -target machine ] [ -t machine ] [ *target machine ]
           [ -tempPruneTime minutes ] [ -tpt minutes ] [ *tempPruneTime minutes ]
           [ -vtsui file ] [ *vtsui file ] [ -help ] [ -h ] [ -? ]

AVAILABILITY  SUNWsymon

DESCRIPTION  symon is the primary user interface to the Solstice SyMON system monitor. Invoking symon brings up the launcher window, from which the seven Solstice SyMON consoles are launched:

• Event Viewer
• Kernel Data Catalog
• Physical View
• Log Viewer
• Logical View
• Process Viewer
• On-line Diagnostics

For further details on the operation of symon please see the Solstice SyMON User's Guide.

OPTIONS  

- colorMap  Use a private color map for the Launcher and Physical View windows to ensure correct colors in the images. May result in colormap flashing of images and of other applications, such as the Netscape browser (default is to use the default colormap).

- cm  Same as -colorMap

*colorMap  Same as -colorMap

- flashDuration  Set time that flashes of the system indicator on the launcher console will last (default is 30 milliseconds).
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-dragthreshold</td>
<td>Sets the mouse drag threshold for Sysmeters (default is 10 pixels).</td>
</tr>
<tr>
<td>*dragthreshold</td>
<td>Same as -dragthreshold</td>
</tr>
<tr>
<td>-fd</td>
<td>Same as -flashDuration</td>
</tr>
<tr>
<td>*flashDuration</td>
<td>Same as -flashDuration</td>
</tr>
<tr>
<td>-flashInterval</td>
<td>Set time interval between flashes of the system indicator on the launcher console (default is 2000 milliseconds).</td>
</tr>
<tr>
<td>-fi</td>
<td>Same as -flashInterval</td>
</tr>
<tr>
<td>*flashInterval</td>
<td>Same as -flashInterval</td>
</tr>
<tr>
<td>-heartbeatInterval</td>
<td>Set the polling time for the heartbeat check for agents (default is 10 intervals).</td>
</tr>
<tr>
<td>-hi</td>
<td>Same as -heartbeatInterval</td>
</tr>
<tr>
<td>-installDir</td>
<td>Set the directory root to examine for tcl files, etc. (default is /opt/SUNWsymon).</td>
</tr>
<tr>
<td>-I</td>
<td>Same as -installDir</td>
</tr>
<tr>
<td>*installDir</td>
<td>Same as -installDir</td>
</tr>
<tr>
<td>-interval</td>
<td>Set the polling interval for agents (default is 10 intervals).</td>
</tr>
<tr>
<td>-i</td>
<td>Same as -interval</td>
</tr>
<tr>
<td>-minWait</td>
<td>Set a minimum wait time between polls/updates (default is 1 second between the end of one poll and the start of the next).</td>
</tr>
<tr>
<td>-mw</td>
<td>Same as -minWait</td>
</tr>
<tr>
<td>-pruneTime</td>
<td>Time after which unchanged data (old processes) is pruned from the sm_krd (Kernel Reader) hierarchy (default is 120 minutes).</td>
</tr>
<tr>
<td>-pt</td>
<td>Same as -pruneTime</td>
</tr>
<tr>
<td>-session</td>
<td>Specifies a Tcl file, which defines the layout and contents of a Solstice SyMON instance. This file is read when Solstice SyMON starts up to restore a previously saved layout.</td>
</tr>
<tr>
<td>-tempPruneTime</td>
<td>Time after which unchanged Config Reader data (board temperature) will be pruned from sm_confgd hierarchy (default is 1440 minutes).</td>
</tr>
<tr>
<td>-tpt</td>
<td>Same as -tempPruneTime</td>
</tr>
<tr>
<td>-target</td>
<td>System to be monitored.</td>
</tr>
<tr>
<td>-t</td>
<td>Same as -target</td>
</tr>
<tr>
<td>-vtsui</td>
<td>Name of SunVTS user interface binary (default is vtsui).</td>
</tr>
<tr>
<td>-help</td>
<td>Listing of arguments.</td>
</tr>
<tr>
<td>-h</td>
<td>Same as -help</td>
</tr>
</tbody>
</table>

1-8 Solaris 2.6 Hardware: 3/98 modified 31 Jan 1997
### ENVIRONMENT

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCL_LIBRARY</td>
<td>Location of the Tcl library.</td>
</tr>
<tr>
<td>XFILESEARCHPATH</td>
<td>Location of the X Files.</td>
</tr>
<tr>
<td>DTAPPSEARCHPATH</td>
<td>Location of the CDE X Defaults files.</td>
</tr>
<tr>
<td>DTDATABASESEARCHPATH</td>
<td>Location of the CDE database files.</td>
</tr>
<tr>
<td>DTHHELPSEARCHPATH</td>
<td>Location of the CDE help files.</td>
</tr>
<tr>
<td>XMICONSEARCHPATH</td>
<td>Location of the symon icons.</td>
</tr>
</tbody>
</table>

### FILES

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>common.tcl</td>
<td>Common Tcl routines for the display.</td>
</tr>
<tr>
<td>cpu_utilization.tcl</td>
<td>Tcl routines to define the chart for CPU utilization.</td>
</tr>
<tr>
<td>memory_usage.tcl</td>
<td>Tcl routines to define the chart for memory usage.</td>
</tr>
<tr>
<td>init.tcl</td>
<td>Tcl routines to initialize symon.</td>
</tr>
<tr>
<td>queue_lengths.tcl</td>
<td>Tcl routines to define the chart for queue lengths.</td>
</tr>
<tr>
<td>sysmeter.tcl</td>
<td>Tcl routines to define the chart for System Meters.</td>
</tr>
</tbody>
</table>

### NOTES

Solstice SyMON uses ASCII-format Tcl files as a means of saving and restoring the state of the program’s GUI. Currently, this feature only works for system meters, the process viewer, and the event viewer. Some Tcl files are provided with the Solstice SyMON product to serve as examples. Normally these Tcl files should be created by using the GUI to configure the desired windows, and then saved by invoking save in a system meter (to save the state of one system meter) or in the kernel data catalog window (to save the state of all system meters).

Symon examines or creates the directory `$HOME/symon` and creates a directory structure there to contain Tcl files that the user has created and links to Tcl files in the official installation. The purpose is that both sets of files may be browsed easily at the same time in a single file selection dialog.

When a Solstice SyMON release is run for the first time by a user, it will create symbolic links in the user’s directory (`$HOME/symon/lib/tcl/C`) that point to any Tcl files in the installation directory (usually `/opt/SUNWsymon/lib/tcl/C`). Thus, any new Tcl files in a new release will be picked up. If the user has files or links in their directory that match the names of files in the official directory, then links will be removed and remade to the official files. User files matching official file names will result in a dialog box in Solstice SyMON that explains the options the user has at that point: Either to keep the local file, to remove it and have Solstice SyMON link to the official version, or to manually merge the two files.

### SEE ALSO

- `sm_configd(1M)`, `sm_confsymon(1M)`, `sm_control(1M)`, `sm_egd(1M)`, `sm_krd(1M)`, `sm_logscand(1M)`, `sm_symond(1M)`, `auth_checker.tcl(4)`, `auth_list.tcl(4)`, `event_gen.tcl(4)`, `logscan.tcl(4)`, `rules.tcl(4)`, `sm_symond.conf(4)`
NAME afbconfig – configure the AFB Graphics Accelerator

SYNOPSIS 
/usr/sbin/afbconfig [-dev device-filename ]
    [ -res video-mode [ now | try ] [ noconfirm | nocheck ] ]
    [ -file machine | system ]
    [ -deflinear true | false ]
    [ -defoverlay true | false ]
    [ -linearorder first | last ]
    [ -overlayorder first | last ]
    [ -expvis enable | disable ]
    [ -sov enable | disable ]
    [ -maxwids n ]
    [ -extovl enable | disable ]
    [ -g gamma-correction-value ]
    [ -gfile gamma-correction-file ]
    [ -propt | -prconf | -defaults ]

AVAILABILITY  SUNWafbcf

DESCRIPTION  afbconfig configures the AFB Graphics Accelerator and some of the X11 window system defaults for AFB.

The first form of afbconfig shown in the synopsis above stores the specified options in the OWconfig file. These options will be used to initialize the AFB device the next time the window system is run on that device. Updating options in the OWconfig file provides persistence of these options across window system sessions and system reboots.

The second and third forms which invoke only the -prconf, -propt, -help, and -res ? options do not update the OWconfig file. Additionally, for the third form all other options are ignored.

Options may be specified for only one AFB device at a time. Specifying options for multiple AFB devices requires multiple invocations of afbconfig.

Only AFB-specific options can be specified through afbconfig. The normal window system options for specifying default depth, default visual class and so forth are still specified as device modifiers on the openwin command line (see the Xsun(1) manual page in the Openwindows Reference Manual).

The user can also specify the OWconfig file that is to be updated. By default, the machine-specific file in the /etc/openwin directory tree is updated. The -file option can be used to specify an alternate file to use. For example, the system-global OWconfig file in the /usr/openwin directory tree can be updated instead.
Both of these standard OWconfig files can only be written by root. Consequently, the `afbconfig` program, which is owned by the root user, always runs with setuid root permission.

**OPTIONS**

- **-dev device-filename**
  Specifies the AFB special file. The default is `/dev/fbs/afb0`.

- **-file machine | system**
  Specifies which OWconfig file to update. If `machine`, the machine-specific OWconfig file in the `/etc/openwin` directory tree is used. If `system`, the global OWconfig file in the `/usr/openwin` directory tree is used. If the file does not exist, it is created.

- **-res video-mode [ now | try [ noconfirm | nocheck ] ]**
  Specifies the video mode used to drive the monitor connected to the specified AFB device.

The format of these built-in video modes is:

widthxheightxrate

where `width` is the screen width in pixels, `height` is the screen height in pixels, and `rate` is the vertical frequency of the screen refresh. The `s` suffix of `960x680x112s` and `960x680x108s` means that these are stereo video modes. The `i` suffix of `640x480x60i` and `768x575x50i` designates interlaced video timing. If absent, non-interlaced timing will be used. As a convenience, `-res` also accepts formats with `'@'` (at sign) in front of the refresh rate instead of `x`. For example: `1280x1024@76`. Note, some video-modes, supported by AFB, may not be supported by the monitor. The list of video-modes supported by the AFB device and the monitor can be obtained by running `afbconfig` with the `-res?` option (the third form shown in the command synopsis above). A list of all possible video-modes supported on AFB is shown below.

1024x768x60
1024x768x70
1024x768x75
1024x768x77
1024x800x84
1152x900x66
1152x900x76
1280x800x76
1280x1024x60
1280x1024x67
1280x1024x76
960x680x112s (Stereo)
960x680x108s (Stereo)
640x480x60
640x480x60i (Interlaced)
768x575x50i (Interlaced)
Symbolic names

For convenience, some of the above video modes have symbolic names defined for them. Instead of the form width x height x rate, one of these names may be supplied as the argument to –res. The meaning of the symbolic name none is that when the window system is run the screen resolution will be the video mode that is currently programmed in the device.

<table>
<thead>
<tr>
<th>Name</th>
<th>Corresponding Video Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>svga</td>
<td>1024x768x60</td>
</tr>
<tr>
<td>1152</td>
<td>1152x900x76</td>
</tr>
<tr>
<td>1280</td>
<td>1280x1024x76</td>
</tr>
<tr>
<td>stereo</td>
<td>960x680x112s</td>
</tr>
<tr>
<td>ntsc</td>
<td>640x480x60i</td>
</tr>
<tr>
<td>pal</td>
<td>768x575x50i</td>
</tr>
<tr>
<td>none</td>
<td>(see text above)</td>
</tr>
</tbody>
</table>

The –res option also accepts additional, optional arguments immediately following the video mode specification. Any or all of these may be present.

- **now** If present, not only will the video mode be updated in the OWconfig file, but the AFB device will be immediately programmed to display this video mode. (This is useful for changing the video mode before starting the window system).

  Note – It is inadvisable to use this suboption with afbconfig while the configured device is being used (e.g. while running the window system); unpredictable results may occur. To run afbconfig with the now suboption, first bring the window system down. If the now suboption is used within a window system session, the video mode will be changed immediately, but the width and height of the affected screen won’t change until the window system is exited and reentered again. In addition, the system may not recognize changes in stereo mode. Consequently, this usage is strongly discouraged.

- **noconfirm** Using the –res option, the user could potentially put the system into an usable state, a state where there is no video output. This can happen if there is ambiguity in the monitor sense codes for the particular code read. To reduce the chance of this, the default behavior of afbconfig is to print a warning message to this effect and to prompt the user to find out if it is okay to continue. The noconfirm option instructs afbconfig to bypass this confirmation and to program the requested video mode anyway. This option is useful when afbconfig is being run from a shell script.

- **nocheck** If present, the normal error checking based on the monitor sense code (described above) will be suspended. The video mode specified by the user will be accepted regardless of whether it is
appropiatete for the currently attached monitor. (This option is useful if a different monitor is to be connected to the AFB device). Use of this option implies noconfirm well.

try

If present, the specified video mode will be programmed on a trial basis. The user will be asked to confirm the video mode by typing 'y' within 10 seconds. Or the user may terminate the trial before 10 seconds are up by typing any character. Any character other than 'y' or carriage return is considered a no and the previous video mode will be restored and afbconfig will not change the video mode in the OWconfig file (other options specified will still take effect). If a carriage return is typed, the user is prompted for a yes or no answer on whether to keep the new video mode. This option implies the now suboption (see the warning note on the now suboption).

AFB possesses two types of visuals: linear and nonlinear. Linear visuals are gamma corrected and nonlinear visuals are not. There are two visuals that have both linear and nonlinear versions: 24-bit TrueColor and 8-bit StaticGray. If true, the default visual is set to the linear visual that satisfies other specified default visual selection options (specifically, the Xsun(1) defdepth and defclass options described in the OpenWindows Reference Manual). If false, or if there is no linear visual that satisfies the other default visual selection options, the non-linear visual specified by these other options will be chosen to be the default.

This option cannot be used when the −defoverlay option is present, because AFB doesn’t possess a linear overlay visual.

−defoverlay true | false

The AFB provides an 8-bit PseudoColor visual whose pixels are disjoint from the rest of the AFB visuals. This is called the overlay visual. Windows created in this visual will not damage windows created in other visuals. The converse, however, is not true. Windows created in other visuals will damage overlay windows. The number of colors available to the windows created using this visual depends on the settings for the extovl option. If the extovl is enabled, extended overlay with 256 opaque color values is available. (refer to the −extovl option). If extovl is disabled, extended overlay is not available and this visual has (256 − maxwids) number of opaque color values (refer to the −maxwids option).

If the value of this option is true, the overlay visual will be made the default visual.

If false, the nonoverlay visual that satisfies the other default visual selection options, such as defdepth and defclass, will be chosen as the default visual. See the Xsun(1) manual page in the OpenWindows Reference Manual.

Whenever −defoverlay true is used, the default depth and class chosen on the openwin command line must be 8-bit PseudoColor. If not, a warning message will be printed and the −defoverlay option will be treated as false.
This option cannot be used when the –delinear option is present, because AFB doesn’t possess a linear overlay visual.

- linearorder first | last
If true, linear visuals will come before their non-linear counterparts on the X11 screen visual list for the AFB screen. If false, the nonlinear visuals will come before the linear ones.

- overlayorder first | last
If true, the depth 8 PseudoColor Overlay visual will come before the non-overlay visual on the X11 screen visual list for the AFB screen. If false, the non-overlay visual will come before the overlay one.

- expvis enable | disable
If enabled, OpenGL Visual Expansion will be activated. Multiple instances of selected visual groups (8-bit PseudoColor, 24-bit TrueColor ... etc) can be found in the screen visual list.

- sov enable | disable
If enabled, the root window’s SERVER_OVERLAY_VISUALS property will be advertised. SOV visuals will be exported and their transparent types, values and layers can be retrieved through this property. If disabled, the SERVER_OVERLAY_VISUALS property will not be defined. SOV visuals will not be exported.

- maxwids n
This option is available only if extovl is disabled. It specifies the maximum number of AFB X channel pixel values that are reserved for use as window IDs (WIDs). The remainder of the pixel values in overlay colormaps are used for normal X11 opaque color pixels.
The reserved WIDs are allocated on a first-come first-serve basis by 3D graphics windows (such as XGL), MBX windows, and windows that have a non-default visual.
The X channel codes 0 to (255 – n) will be opaque color pixels. The X channel codes (255 – n + 1) to 255 will be reserved for use as WIDs. Legal values: 1, 2, 4, 8, 16, 32, 64

- extovl enable | disable
If enabled, extended overlay is available. The overlay visuals will have 256 opaque colors. The SOV visuals will have 255 opaque colors and 1 transparent color. Also, this option enables hardware supported transparency, thus provides better performance for windows using the SOV visuals.

- g gamma-correction value
This option allows changing the gamma correction value. All linear visuals provide gamma correction. By default the gamma correction value is 2.22. Any value less than zero is illegal.
This option can be used while the window system is running. Changing the gamma correction value will affect all the windows being displayed using the linear visuals.
The option `-gfile gamma-correction file` loads gamma correction table from the specified file. This file should be formatted to provide the gamma correction values for R, G and B channels on each line. Each of these values should be in hexadecimal format and separated from each other by at least 1 space. Also, this file should provide 256 such triplets. An example of this file is as follows:

```
0x00 0x00 0x00
0x01 0x01 0x01
0x02 0x02 0x02
...
...
0xff 0xff 0xff
```

Using this option, the gamma correction table can be loaded while the window system is running. The new gamma correction will affect all the windows being displayed using the linear visuals. Note, when gamma correction is being done using a user-specified table, the gamma correction value is undefined.

By default, the window system assumes a gamma correction value of 2.22 and loads the gamma table it creates corresponding to this value.

The option `-defaults` resets all option values to their default values.

The option `-propt` prints the current values of all AFB options in the OWconfig file specified by the `-file` option for the device specified by the `-dev` option. Prints the values of options as they will be in the OWconfig file after the call to `afbconfig` completes. This is a typical display:

```
--- OpenWindows Configuration for /dev/fbs/afb0 ---
OWconfig: machine
Video Mode: 1280x1024x76
Default Visual: Non-Linear Normal Visual
Visual Ordering: Linear Visuals are last
          Overlay Visuals are last
OpenGL Visual Expansion: enabled
Server Overlay Visuals: enabled
Extended Overlay: enabled
Underlay WIDs: 64 (not configurable)
Overlay WIDs: 4 (not configurable)
Gamma Correction Value: 2.220000
Gamma Correction Table: Available
```

The option `-prconf` prints the AFB hardware configuration. This is a typical display:

```
--- Hardware Configuration for /dev/fbs/afb0 ---
Type: double-buffered AFB with Z-buffer
```

modified 28 Mar 1996
Solaris 2.6 Hardware: 3/98
Board: rev 0 (Horizontal)
PROM Information: @(#)afb.fth xx/xx
FBC: version 0x101df06d
DAC: Brooktree 9070, version 1 (Pac2)
3DRAM: Mitsubishi 130a, version x
EDID Data: Available - EDID version 1 revision x
Monitor Sense ID: 4 (Sun 37x29cm RGB color monitor)
Monitor possible resolutions: 1024x768x77, 1024x800x84, 1152x900x66,
1152x900x76, 1280x1024x67, 1280x1024x76, 960x680x112s, 960x680x108s
Current resolution setting: 1280x1024x76

--help  Prints a list of the afbconfig command line options, along with a brief explanation of each.

DEFAULTS  For a given invocation of afbconfig command line if an option does not appear on the command line, the corresponding OWconfig option is not updated; it retains its previous value.

When the window system is run, if an AFB option has never been specified via afbconfig, a default value is used. The option defaults are as follows:

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>-dev</td>
<td>/dev/fbs/afb0</td>
</tr>
<tr>
<td>-file</td>
<td>machine</td>
</tr>
<tr>
<td>-res</td>
<td>none</td>
</tr>
<tr>
<td>-deflinear</td>
<td>false</td>
</tr>
<tr>
<td>-defoverlay</td>
<td>false</td>
</tr>
<tr>
<td>-linearorder</td>
<td>last</td>
</tr>
<tr>
<td>-overlayorder</td>
<td>last</td>
</tr>
<tr>
<td>-expvis</td>
<td>enabled</td>
</tr>
<tr>
<td>-sov</td>
<td>enabled</td>
</tr>
<tr>
<td>-maxwids</td>
<td>32</td>
</tr>
<tr>
<td>-extovl</td>
<td>enabled</td>
</tr>
<tr>
<td>-g</td>
<td>2.22</td>
</tr>
</tbody>
</table>

The default for the -res option of none means that when the window system is run the screen resolution will be the video mode that is currently programmed in the device.

Note  This provides compatibility for users who are used to specifying the device resolution through the PROM. On some devices (e.g. GX) this is the only way of specifying the video mode. This means that the PROM ultimately determines the default AFB video mode.

EXAMPLES  The following example switches the monitor type to the resolution of 1280 x 1024 at 76 Hz:

eexample% /usr/sbin/afbconfig -res 1280x1024x76
## Files

<table>
<thead>
<tr>
<th>FILES</th>
<th>device special file</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/fbs/afb0</td>
<td>device special file</td>
</tr>
</tbody>
</table>

## See Also

<table>
<thead>
<tr>
<th>SEE ALSO</th>
<th>mmap(2), fbio(7I), afb(7D)</th>
</tr>
</thead>
</table>

modified 28 Mar 1996

Solaris 2.6 Hardware: 3/98
NAME
cfgadm – configuration administration

SYNOPSIS
/usr/sbin/cfgadm [-f] [-y] [-n] [-v] [-o hardware_options] [-x hardware_function ap_id [ap_id...]]
/usr/sbin/cfgadm [-v] [-s listing_options] [-o hardware_options] [-l [ap_id | ap_type...]]
/usr/sbin/cfgadm [-v] [-o hardware_options] [-t ap_id [ap_id...]]

AVAILABILITY
SUNWcsu

DESCRIPTION
The cfgadm command provides configuration administration operations on dynamically reconﬁgurable hardware resources. These operations include displaying status, (−l), initiating testing, (−t), invoking conﬁguration state changes, (−c), and invoking hardware speciﬁc functions that do not change state, (−x). Conﬁguration administration is performed at attachment points which are places where system software supports dynamic reconﬁguration of hardware resources during continued operation of Solaris.

Conﬁguration administration makes a distinction between hardware resources that are physically present in the machine and hardware resources that are conﬁgured and visible to Solaris. The nature of conﬁguration administration functions are hardware speciﬁc, and are performed by calling hardware speciﬁc libraries.

Conﬁguration administration operates on an attachment point. Hardware resources located at attachment points may or may not be physically replaceable during system operation, but are dynamically reconﬁgurable via the conﬁguration administration interfaces. Attachment points are represented in the device tree by a unique nodetype.

An attachment point deﬁnes two unique elements, which are distinct from the hardware resources that exist beyond the attachment point. The two elements of an attachment point are a receptacle and an occupant. Physical insertion or removal of hardware resources occurs at attachment points and results in a receptacle gaining or losing an occupant. Conﬁguration administration supports the physical insertion and removal operations as well as other conﬁguration administration functions at an attachment point.

Attachment points have associated state and condition information. The conﬁguration administration interfaces provide control for transitioning attachment point states. A receptacle can exist in one of four states: empty, prepped, disconnected or connected, while an occupant exist in one of two states: conﬁgured or unconfigured.

A receptacle must provide the empty state, which is the normal state of a receptacle when the attachment point has no occupants. A receptacle may provide the prepped state which prepares the hardware for the physical insertion or removal of occupants. This involves hardware speciﬁc functions that may temporarily suspend activity on portions of the running system. A receptacle may provide the disconnected state if it has the capability of isolating its occupants from normal system access. Typically this state is used for various hardware speciﬁc testing prior to bringing the occupant’s resources into full use by the system, or as a step in preparing an occupant for...
physical removal or reconfiguration. A receptacle in the disconnected state isolates its occupant from the system as much as its hardware allows, but may provide access for testing and setup. A receptacle must provide the **connected** state, which allows normal access to hardware resources contained on any occupants. The connected state is the normal state of a receptacle that contains an occupant and that is not currently undergoing configuration administration operations.

The hardware resources contained on an occupant in the **unconfigured** state are not represented by normal Solaris data structures (such as device tree nodes) and are thus not available for use by Solaris. Operations allowed on an unconfigured occupant are limited to configuration administration operations. The hardware resources of an occupant in the **configured** state are represented by normal Solaris data structures and thus some or all of those hardware resources may be in use by Solaris. An occupant is required to provide both the **configured** and **unconfigured** states.

An attachment point may be in one of **five conditions**: **unknown**, **ok**, **failing**, **failed** and **unusable**. An attachment point can enter the system in any condition depending upon results of power-on tests and non-volatile record keeping.

An attachment point with an occupant in the **configured** state will be in one of four conditions: **unknown**, **ok**, **failing** or **failed**. If the condition is not **failing** or **failed** an attachment point may change to **failing** during the course of operation if a hardware dependent recoverable error threshold is exceeded. If the condition is not **failed** an attachment point may change to **failed** during operation as a result of an unrecoverable error.

An attachment point with an occupant in the **unconfigured** state can be in any of the defined conditions. The condition of an attachment point with an **unconfigured** occupant may decay from **ok** to **unknown** after a machine dependent time threshold. Initiating a test function will change the attachment point’s condition to **ok**, **failing** or **failed** depending on the outcome of the test. An attachment point that does not provide a test function may leave the attachment point in the **unknown** condition. If a test is interrupted, the attachment point’s condition may be set to the previous condition, **unknown** or **failed**. An attachment point in the **unknown**, **ok**, **failing** or **failed** conditions can be re-tested.

An attachment point may exist in the **unusable** condition for a variety of reasons, such as inadequate power or cooling for the receptacle, an occupant that is unidentifiable, unsupported, incorrectly configured, etc. An attachment point in the **unusable** condition can never be used by the system. It will typically remain in this condition until the physical cause is remedied.

An attachment point also maintains an activity field that indicates when the condition is being reevaluated, **reevaluating condition**, is in the process of a state change, **state changing**, or executing a private function, **executing private**.

Attachment points are referred to using hardware specific identifiers (**ap_id**) that are related to the type and location of the attachment points in the system device hierarchy. An **ap_id** may not be ambiguous, it must identify a single attachment point. Two types of **ap_id** specifications are supported: physical and logical. A physical **ap_id**
contains a fully specified pathname, while a logical \textit{ap\_id} contains a shorthand notation to identify an attachment point in a more user-friendly way. Both types of \textit{ap\_ids} share a common format, with the name and instance forming the initial part followed by a "." and then the specific attachment point. For example, a receptacle representing a system’s backplane slot number 7 might have a physical \textit{ap\_id} of \\
/central/fhc/sysctrl:slot7 while the logical \textit{ap\_id} might be \texttt{system:slot7}. Another example, the third receptacle on the second PCI I/O bus on a system might have a logical \textit{ap\_id} of \texttt{pci2:plug3}.

The \texttt{cfgadm} command parses an \textit{ap\_id} and uses the name portion to locate and dynamically load the hardware specific library that supports that type of attachment point. The \textit{ap\_id} is passed on to the hardware specific library to perform operations. The hardware specific library validates that the \textit{ap\_id} is complete and identifies a single attachment point to operate on.

An \textit{ap\_type} is a partial form of an \textit{ap\_id} that may be ambiguous and not specify a particular attachment point. The \textit{ap\_type} is used by the list function to allow listing of all attachment points of the same type. It consists of the name portion of an \textit{ap\_id} and may omit the instance, the colon separator and the specific attachment point identifier. For example, a logical \textit{ap\_id} of \texttt{pci} would show all attachment points supported by the library identified by the name \texttt{pci}.

The \texttt{cfgadm} command parses an \textit{ap\_type} and uses the name portion to locate and dynamically load the hardware specific library that supports attachment points of that type. The \textit{ap\_type} is passed to the hardware specific library to perform listing operations.

The \texttt{cfgadm} command interacts primarily with hardware dependent functions contained in hardware specific libraries and thus its behavior is hardware dependent.

For each configuration administration operation a service interruption may be required. Should the completion of the function requested require a noticeable (to interactive users) service interruption, a prompt will be output on the standard error output for confirmation on the standard input before the function is started. Confirmation can be overridden using the \texttt{-y} or \texttt{-n} flags to always answer yes or no respectively. Hardware specific options, such as test level, are supplied as sub-options using the \texttt{-o} flag.

Operations that change the state of the system configuration are audited by the system log daemon \texttt{syslogd(1M)}.

The arguments for this command conform to the \texttt{getopt(3C)} and \texttt{getsubopt(3C)} syntax convention.

\textbf{OPTIONS}
\begin{itemize}
\item \texttt{-c \textit{function}}
\end{itemize}

Performs the specified state change \textit{function} on the attachment point specified by \textit{ap\_id}. The available \textit{functions} are \texttt{prep}, \texttt{load}, \texttt{unload}, \texttt{resume}, \texttt{disconnect}, \texttt{connect}, \texttt{configure} and \texttt{unconfigure}. These functions cause state transitions at the attachment point by calling hardware specific library routines.

The \texttt{prep} function performs hardware specific operations to prepare a
receptacle for the physical addition or removal of an occupant. Various hardware specific safeguards may cause this function to fail and set the receptacle condition to **unusable**.

The **load** function performs operations to activate hardware supplied insertion hardware that will perform the physical addition of an occupant. Various hardware specific errors may cause this function to fail and set the receptacle condition to **unusable**.

The **unload** function performs operations to activate hardware supplied removal hardware that will perform the physical removal of an occupant. Various hardware specific errors may cause this function to fail and set the receptacle condition to unusable.

The **resume** function performs hardware specific operations to transition a receptacle from the **prepped** state to the **empty** state when all occupants have been removed, or to the **disconnected** state, if provided, when an occupant has been added. If the attachment point does not support the **disconnected** state the transition will be to **connected**.

The **disconnect** function performs hardware specific operations to put a receptacle in the disconnected state, which may prevent an occupant from operating in a normal fashion through the receptacle.

The **connect** function performs hardware specific operations to put the receptacle in the **connected** state, which allows an occupant to operate in a normal fashion through the receptacle.

The **configure** function performs hardware specific operations that allow an occupant’s hardware resources to be usable by Solaris. Occupants that are configured are part of the system configuration and will be available for manipulation by Solaris device manipulation maintenance commands (eg: **psradm**(1M), **mount**(1M), **ifconfig**(1M)).

The **unconfigure** function performs hardware specific operations that logically remove an occupant’s hardware resources from the system. The occupant must currently be configured and its hardware resources must not be in use by Solaris.

State transition functions may fail due to the condition of the attachment point or other hardware dependent considerations. All state change functions in the direction of adding resources, are passed onto the hardware specific library when the attachment point is in the **ok** or **unknown** condition. All other conditions require the use of the force option to allow functions in the direction of adding resources, to be passed on to the hardware specific library. Attachment point condition does not gate transitions related to removal of hardware resources from the system. Hardware specific libraries may reject state change functions if the attachment point is in the **unknown** condition.

The condition of an attachment point is not necessarily changed by the state change functions, however errors during state change operations can change the attachment point condition. An attempt to override a condition and force a state change that
would otherwise fail can be made by specifying the force flag (`-f`). Hardware specific
safety and integrity checks may prevent the force flag from having any effect.

`-f` Force the specified action to occur. Typically, this is a hardware dependent
override of a safety feature. Forcing a state change operation may allow use of
the hardware resources of occupant that is not in the `ok` or `unknown`
conditions, at the discretion of any hardware dependent safety checks.

`-l` List the state and condition of attachment points. If any `ap_types` are specified,
the listing will be limited to attachment points of those types. If any `ap_ids` are
specified the listing is limited to those particular attachment points. Invoking
the `cfgadm` command without one of the action flags is equivalent to `-l`
without an `ap_id` or an `ap_type` argument. The format of the display is con-
trolled by the `-v` and `-s` options.

`-n` Suppress any interactive confirmation and assume that the answer is `no`. If
neither `-n` or `-y` is specified, interactive confirmation is obtained through the
standard error output and the standard input. If either of these standard chan-
nels does not correspond to a terminal (as determined by `isatty(3C)`) then the
`-n` option is assumed.

`-o hardware_options` Supply hardware specific options to the main command option. The format
and content of the hardware option string is completely hardware specific.
The option string `hardware_options` conforms to the `getsubopt(3C)` syntax con-
vention.

`-s listing_options` Supply listing options to the list (`-l`) command. The option string
`listing_options` conforms to the `getsubopt(3C)` syntax convention. The sub-
opions are used to control the order of the listing (`sort=field_spec`), the data
that is displayed (`cols=field_spec` and `cols2=field_spec`), the column delimiter
(`delim=string`) and whether to suppress column headings (`noheadings`). A
`field_spec` is one or more `data-fields` concatenated using `:` (colon) as in `data-
field: data-field`. A `data-field` is one of `ap_id`, `physid`, `r_state`, `o_state`,
`condition`, `activity`, `type`, `status_time`, `status_time_p` and `info`. The
`status_time_p` field is a parsable version of the `status_time` field. The order of
the fields in `field_spec` is significant: For the `sort` sub-option, the first field
given is the primary sort key. For the `cols` and `cols2` sub-options, the fields are
printed in the order requested. The fields in `cols` are always printed. The
fields in `cols2` are printed on a second line only if specified. The order of sort-
ing on a `data-field` may be reversed by placing a `-` (minus) before the `data-field`
name within the `field_sec` for the `sort` sub-option. The default value for `sort` is
`ap_id`. The defaults values for `cols` and `cols2` depend on whether the `-v`
option is given: Without it `cols` is `ap_id:r_state:o_state:condition` and `cols2` is
not set. With `-v cols` is `ap_id:r_state:o_state:condition:info` and `cols2` is
`status type:activity:physid:`. The default value for `delim` is a single space.
The value of `delim` may be a string of arbitrary length. The delimiter cannot
include comma character, see `getsubopt(3C)`. These listing options may be
used to create parsable output - see NOTES below.

−t Perform a test of one or more attachment points. The test function is used to re-evaluate the condition of the attachment point. Without a test level specifier in *hardware_options*, the fastest test that will identify hard faults will be used.

More comprehensive tests are hardware specific and are selected using the *hardware_options*.

The results of the test will be used to update the condition of the specified occupant to either **ok** if no faults are found, **failing** if recoverable faults are found or **failed** if any unrecoverable faults are found.

If a test is interrupted, the attachment point's condition may be restored to its previous value or set to **unknown** if no errors were found or **failing** if only recoverable errors were found or to **failed** if any unrecoverable errors were found. The attachment point should only be set to **ok** when upon normal completion of testing with no errors.

−v Verbose option. For the −c, −t and −x options output a message giving the results of each attempted operation. For the −l option output full information for each attachment point.

−x *hardware_function*

Performs hardware specific functions. These private hardware specific functions are not allowed to change the state of a receptacle or occupant. Attachment point conditions may change as the result of errors encountered during private hardware specific functions. The format and content of the *hardware_function* string is completely hardware specific. The option string *hardware_function* conforms to the *getsubopt*(3C) syntax convention.

−y Suppress any interactive confirmation and assume that the answer is yes.

**USAGE**
The required privileges to use this command are hardware dependent. Typically, a default system configuration will restrict all but the list option to the superuser.

**EXAMPLES**
List current configurable hardware information

```
example# cfgadm

<table>
<thead>
<tr>
<th>Ap_Id</th>
<th>Receptacle</th>
<th>Occupant</th>
<th>Cond</th>
</tr>
</thead>
<tbody>
<tr>
<td>system:slot0</td>
<td>connected</td>
<td>configured</td>
<td>ok</td>
</tr>
<tr>
<td>system:slot1</td>
<td>connected</td>
<td>configured</td>
<td>ok</td>
</tr>
<tr>
<td>system:slot2</td>
<td>connected</td>
<td>configured</td>
<td>ok</td>
</tr>
<tr>
<td>system:slot3</td>
<td>connected</td>
<td>unconfigured</td>
<td>unknown</td>
</tr>
<tr>
<td>system:slot4</td>
<td>connected</td>
<td>configured</td>
<td>failing</td>
</tr>
<tr>
<td>system:slot5</td>
<td>connected</td>
<td>configured</td>
<td>ok</td>
</tr>
<tr>
<td>system:slot6</td>
<td>disconnected</td>
<td>unconfigured</td>
<td>unusable</td>
</tr>
<tr>
<td>system:slot7</td>
<td>empty</td>
<td>unconfigured</td>
<td></td>
</tr>
</tbody>
</table>
```

List current configurable hardware information in verbose mode

```
example# cfgadm -v -l system

Status of system configuration at Wed Nov 13 17:26:17 PST 1996
```

modified 5 Nov 1997 Solaris 2.6 Hardware: 3/98 1M-23
**cfgadm (1M)**

<table>
<thead>
<tr>
<th>Ap_Id</th>
<th>Receptacle When</th>
<th>Occupant Type</th>
<th>Cond Activity</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>system:slot0</td>
<td>connected</td>
<td>configured</td>
<td>ok</td>
<td>SUNW, UltraSPARC, 168 MHz /central/f/hc/sysctrl:slot0</td>
</tr>
<tr>
<td></td>
<td>Nov 5</td>
<td>CPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>system:slot1</td>
<td>connected</td>
<td>configured</td>
<td>ok</td>
<td>512mb, 2 way interleaved /central/f/hc/sysctrl:slot1</td>
</tr>
<tr>
<td></td>
<td>Nov 5</td>
<td>MEMORY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>system:slot2</td>
<td>connected</td>
<td>configured</td>
<td>ok</td>
<td>PCI                       /central/f/hc/sysctrl:slot2</td>
</tr>
<tr>
<td></td>
<td>Nov 5</td>
<td>IO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>system:slot3</td>
<td>connected</td>
<td>unconfigured</td>
<td>failing</td>
<td>512mb, 2 way interleaved /central/f/hc/sysctrl:slot3</td>
</tr>
<tr>
<td></td>
<td>Nov 5</td>
<td>MEMORY</td>
<td>reevaluating_condition</td>
<td></td>
</tr>
<tr>
<td>system:slot4</td>
<td>connected</td>
<td>configured</td>
<td>failing</td>
<td>512mb, 2 way interleaved /central/f/hc/sysctrl:slot4</td>
</tr>
<tr>
<td></td>
<td>Nov 5</td>
<td>MEMORY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>system:slot5</td>
<td>connected</td>
<td>configured</td>
<td>ok</td>
<td>PCI                       /central/f/hc/sysctrl:slot5</td>
</tr>
<tr>
<td></td>
<td>Nov 5</td>
<td>IO</td>
<td>state_changing</td>
<td></td>
</tr>
<tr>
<td>system:slot6</td>
<td>disconnected</td>
<td>unconfigured</td>
<td>unusable</td>
<td>unsupported option /central/f/hc/sysctrl:slot6</td>
</tr>
<tr>
<td></td>
<td>Nov 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>system:slot7</td>
<td>empty</td>
<td>unconfigured</td>
<td>ok</td>
<td>/central/f/hc/sysctrl:slot7</td>
</tr>
<tr>
<td></td>
<td>Nov 5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Test two occupants using the hardware specific extended test:
```bash
eample# cfgadm -v -o extended -t system:slot3 system:slot5
testing attachment point system:slot3 ... ok
testing attachment point system:slot5 ... ok
```

Configure an occupant in the failing state to the system using the force option:
```bash
eample# cfgadm -f -c configure system:slot3
```

Unconfigure an occupant from the system:
```bash
eample# cfgadm -c unconfigure system:slot4
```

**ENVIRONMENT**

See `environ(5)` for descriptions of the following environment variables that affect the execution of `cfgadm`: `LC_TIME`, `LC_MESSAGES`, `NLSPATH` and `TZ`.

**LC_MESSAGES**

Determines how `cfgadm` displays column headings and error messages. Listing output data is not affected by the setting of this variable.

**LC_TIME**

Determines how `cfgadm` displays human readable status changed time (`status_time`).

**TZ**

Specifies the timezone used when converting the status changed time. This applies to both the human readable (`status_time`) and parsable (`status_time_p`) formats.

**EXIT STATUS**

The following exit values are returned:

<table>
<thead>
<tr>
<th>Exit Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The operation completed successfully.</td>
</tr>
<tr>
<td>1</td>
<td>The operation failed.</td>
</tr>
</tbody>
</table>

1M-24 Solaris 2.6 Hardware: 3/98 modified 5 Nov 1997
2  Configuration administration not supported on specified target.
3  Usage error.

SEE ALSO  prtdiag(1M), psradm(1M), syslogd(1M), getopt(3C), getsubopt(3C), config_admin(3X).

DIAGNOSTICS  Diagnostic messages will appear on the standard error output. Other than options and
usage errors, the following may be seen:

cfgadm: Configuration administration not supported on ap_id

cfgadm: No library found for ap_id

cfgadm: ap_id is ambiguous

cfgadm: operation: Insufficient privileges

cfgadm: attachment point is busy, try again

cfgadm: System is busy, try again

cfgadm: operation: Operation requires a service interruption

cfgadm: operation: Data error: error_text

cfgadm: operation: Hardware specific failure: error_text

See config_admin(3X) for additional error message detail.

NOTES  Hardware resources will enter the unconfigured pool in a hardware specific manner. This can occur at various times such as: system initialization or as a result of an unconfigure operation. An occupant that is in the unconfigured state is not available for use by the system until specific intervention occurs. This intervention may be manifested as an operator initiated command or it may be via an automatic configuring mechanism.

The listing option of the cfgadm command can be used to provide parsable input for another command, for example within a shell script. The -s option can be used to select the fields required and suppress the column headings. The following fields will always produce parsable output: ap_id, physid, r_state, o_state, condition, activity, status_time_p and type. Parsable output will never have white-space characters embedded in the field value. Here is a shell script fragment to find the first good unconfigured occupant of type CPU.

```bash
found=  

cfgadm -l -s "noheadings,cols=ap_id:r_state:condition:type" | \ 
while read ap_id r_state cond type
do
  if [ "$r_state" = unconfigured -a "$cond" = ok -a "$type" = CPU ]
    then
      if [ -z "$found" ]
        then
          found=$ap_id
        fi
    fi
```

modified 5 Nov 1997  Solaris 2.6 Hardware: 3/98  1M-25
done
if [ -n "$found" ]
then
    echo "Found CPU $found"
fi

The format of the parsable time field (status_time_p) is YYYYMMDDhhmcss, giving the year, month, day, hour, minute and second in a form suitable for string comparison.

Reference should be made to the hardware specific documentation for details of System Configuration Administration support.
**NAME**
cvcd – virtual console daemon

**DESCRIPTION**
cvcd is a server that resides on an Enterprise 10000 host or domain. It accepts connections from `netcon_server(1M)` on an SSP to create a Network Console Window on that SSP. The Network Console Window is able to read data from, and possibly send data to, the host or domain. This process takes place via the SSP command `netcon(1M)`. See `netcon_server(1M)` and `netcon(1M)` in *man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands*.

When you execute `netcon(1M)` in an SSP Window, `netcon_server(1M)` connects with the cvcd daemon running on the host or domain specified in the SSP’s SUNW_HOSTNAME environment variable, and the window becomes a Host Console Window.

The console session ends when you exit the session, `netcon_server` terminates, or a network failure occurs. If cvcd dies, netcon gets data from JTAG through the control board.

cvcd is normally started during boot. Only one cvcd process at a time can run on the host.

Caution: cvcd uses the file `ssphostname`, which resides on the host. If the SSP has been renamed, ssphostname must be edited to reflect that change.

**SEE ALSO**
*Ultra Enterprise 10000 SSP User’s Guide*

cvc(7), cvcredir(7), netcon(1M), netcon_server(1M), services(4)
NAME  

\texttt{dr\_daemon} – dynamic reconfiguration daemon

SYNOPSIS  

\texttt{dr\_daemon \([-a]\)}

DESCRIPTION  

The \texttt{dr\_daemon} is an RPC program that provides the interface to the Dynamic Reconfiguration (DR) driver, \texttt{/dev/dr}. The Hostview and DR applications provide the user interface to DR. See \texttt{hostview(1M)} in \textit{man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands} and \texttt{dr(1M)} in \textit{man Pages(1M): DR Administration Commands}.

OPTIONS  

\texttt{-a} Disable communications with the Alternate Pathing daemon. See \texttt{ap\_daemon(1M)} in \textit{man Pages(1M): Alternate Pathing Administration Commands}.

Configuration Information  

The \texttt{/usr/platform/sun4u1/sbin/dr\_daemon} RPC program name is DRPROG, its RPC program number is 300326, and its underlying protocol is TCP. It is invoked as an inetd server using the TCP transport. The UID required for access to the daemon is ssp. This UID can be a non-login UID.

The entry for the daemon in the \texttt{/etc/inetd.conf} file is:

\begin{verbatim}
300326/4  tli rpc/tcp wait root /usr/platform/sun4u1/sbin/dr_daemon dr_daemon
\end{verbatim}

The daemon’s only clients are Hostview and DR. Hostview provides a GUI interface; \texttt{dr(1M)} is a command-line interface for non-windowing environments. The DR daemon uses \texttt{syslog(3)} to report status and error messages, which are logged with the LOG_DAEMON facility and the LOG_ERR and LOG_NOTICE priorities.

The \texttt{dr\_daemon} communicates via RPC with the Alternate Pathing (AP) daemon (see \texttt{ap\_daemon(1M)} in \textit{man Pages(1M): Alternate Pathing Administration Commands}) to notify the AP software when controllers are attached to and detached from the system, or to gather information about the system configuration.

SEE ALSO  

\textit{Dynamic Reconfiguration User’s Guide}  
\textit{Alternate Pathing User’s Guide}

\texttt{dr(7)} in this reference manual

\texttt{ap(1M), ap\_daemon(1M)} in \textit{man Pages(1M): Alternate Pathing Administration Commands}

\texttt{dr(1M)} in \textit{man Pages(1M): DR Administration Commands}

\texttt{hostview(1M), hpost(1M)} in \textit{man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands}

\texttt{add\_drv(1M), drvconfig(1M), devlinks(1M), disks(1M), inetd(1M), ports(1M), tapes(1M), prtconf(1M), syslog(3)} in \textit{man Pages(1M): System Administration Commands}
NAME
luxadm – administration program for the Sun Enterprise Network Array (SENA), RSM and SPARCstorage Array (SSA) subsystems

SYNOPSIS
luxadm [ options ... ] subcommand [ options ... ] enclosure [ ,dev ] | pathname ...

DESCRIPTION
The luxadm program is an administrative command that manages the SENA, RSM, and SPARCstorage Array subsystems. luxadm performs a variety of control and query tasks depending on the command line arguments and options used.

The command line must contain a subcommand. The command line may also contain options, usually at least one enclosure name or pathname, and other parameters depending on the subcommand. You need specify only as many characters as are required to uniquely identify a subcommand.

Specify the device that a subcommand interacts with by entering a pathname. For the SENA subsystem, a disk device or enclosure services controller may instead be specified by entering the World Wide Name (WWN) for the device or a port to the device. The device may also be specified by entering the name of the SENA enclosure, and an optional identifier for the particular device in the enclosure.

Pathname
Specify the device or controller by either a complete physical pathname or a complete logical pathname.

For SENA, a typical physical pathname for a device is:

/devices/sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w2200002037000f96,0:a,raw
or
/devices/io-unit@f,e0200000/sbi@0,0/SUNW,socal@2,0/sf@0,0/ssd@34,0:a,raw

For all SENA IBs (Interface Boards) on the system, a logical link to the physical paths is kept in the directory /dev/es. An example of a logical link is /dev/es/ses0.

For SENA, the WWN may be used in place of the pathname to select a device or SENA subsystem IB. The WWN is a unique 16 hexadecimal digit value that specifies either the port used to access the device or the device itself. A typical WWN value is:

2200002037000f96

See NOTES for more information on the WWN formats.

For the SPARCstorage Array controller, a typical physical pathname is:

/devices/.../SUNW,soc@3,0/SUNW,pln@axxxxxxx,xxxxxxxxx:ctrl
whereas, a typical physical pathname for an RSM controller might be:

/devices/sbus@1f,0/QLGC,isp@1,10000:devctl

In order to make it easier to address the SPARCstorage Array or RSM controller, a logical pathname of the form cN is supported, where N is the logical controller number. luxadm uses the cN name to find an entry in the /dev/rdsk directory of a disk that is attached to the SPARCstorage Array or RSM controller. The /dev/rdsk entry is then used to determine the physical name of the SPARCstorage Array or RSM controller.
For a SPARCstorage Array disk, a typical physical pathname is:

```
/devices/.../SUNW,soc@3,0/SUNW,pln@xxxxxxx,xxxxxxx/ssd@0,0:c,raw
```

and a typical logical pathname is:

```
/dev/rdsk/c1t0d0s2
```

For an RSM a typical physical pathname might be:

```
/devices/sbus@1f,0/QLGC,isp@1,10000/sd@8,0:c,raw
```

and a typical logical pathname might be:

```
/dev/rdsk/c2t8d0s2
```

**Enclosure**

For SENA, a device may be identified by its enclosure name and slotname:

```
box_name[,f slot_number]
box_name[,r slot_number]
```

*box_name* is the name of the SENA enclosure, as specified by the *enclosure_name* subcommand. When used without the optional *slot_number* parameter, the *box_name* identifies the SENA subsystem IB.

*f* or *r* specifies the front or rear slots in the SENA enclosure.

*slot_number* specifies the slot number of the device in the SENA enclosure, 0-6 or 0-10.

See *disks*(1M) and *devlinks*(1M) for additional information on logical names for disks and subsystems.

**OPTIONS**

The following options are supported by all subcommands:

- `-e` Expert mode. This option is not recommended for the novice user.
- `-v` Verbose mode.

Options that are specific to particular subcommands are described with the subcommand in the *USAGE* section.

**OPERANDS**

The following operands are supported:

- `enclosure` The box_name of the SENA.
- `pathname` The logical or physical path of a SENA IB, SPARCstorage Array or RSM controller (cN name) or disk device. *pathname* can also be the WWN of a SENA IB or SENA disk.

**USAGE**

**Subcommands**

```
display enclosure[,dev]... | pathname...
display -p pathname...
display -r enclosure[,dev]... | pathname...
display -v enclosure[,dev]... | pathname...
```

Displays enclosure or device specific data.

Subsystem data consists of enclosure environmental sense information and

---

**Modified**

Solaris 2.6 Hardware: 3/98

modified 20 Oct 1997
status for all subsystem devices, including disks.

Disk data consists of inquiry, capacity, and configuration information.

- \( \mathbf{p} \) Displays performance information for the device or subsystem specified by \textit{pathname}. This option only applies to subsystems that accumulate performance information.

- \( \mathbf{r} \) Displays error information for the device specified by the pathname, or, if the path is a SENA, for all devices on the loop. The \( \mathbf{r} \) option only applies to SENA subsystems.

- \( \mathbf{v} \) Displays in verbose mode, including mode sense data.

\textbf{probe [ } \mathbf{- p} \textbf{ ]}

Finds and displays information about all attached SENA subsystems, including the logical pathname, the WWNs, and enclosure names. This subcommand warns the user if it finds different SENAs with the same enclosure names.

- \( \mathbf{p} \) Includes the physical pathname in the display.

\textbf{download [ } \mathbf{- s} \textbf{ ] [ } \mathbf{- w} \text{ WWN } \text{ ] [ } \mathbf{- f} \text{ filename} \_\text{path} \text{ ] enclosure\ldots } \text{ | pathname\ldots } \text{ ]}

Download the prom image pointed to by \textit{filename} \_\textit{path} to the SENA subsystem Interface Board unit or the SPARCstorage Array controllers specified by the enclosure or pathname. The SPARCstorage Array must be reset in order to use the downloaded code.

When the SENA’s download is complete, the SENA will be reset and the downloaded code executed. If no filename is specified, the default prom image will be used. The default prom image for the SPARCstorage Array controller is in \texttt{usr/lib/firmware/ssa/ssafirmware}. The default prom image for the SENA is in the directory \texttt{usr/lib/locale/C/LC_MESSAGES} and is named \texttt{ibfirmware}. The SENA firmware is language dependent so the \texttt{LANG} environment variable is used to find the directory that contains the firmware. The default directory is \texttt{C}.

- \( \mathbf{- s} \) Save. The \( \mathbf{- s} \) option is used to save the downloaded firmware in the FEPROM. If \( \mathbf{- s} \) is not specified, the downloaded firmware will not be saved across power cycles. The \( \mathbf{- s} \) option does not apply to the SPARCstorage Array controller as it \textit{always} writes the downloaded firmware into the FEPROM. When using the \( \mathbf{- s} \) option, the \texttt{download} subcommand modifies the FEPROM on the subsystem and should be used with caution.

- \( \mathbf{- w} \) WWN

Change the SPARCstorage Array controller’s World Wide Name. \texttt{WWN} is a 12-digit hex number; leading zeros are required. The \( \mathbf{- w} \) option applies only to the SPARCstorage Array. The new SPARCstorage Array controller’s image will have the least significant 6 bytes of the 8-byte World Wide Name modified to \texttt{WWN}.

\textbf{enclosure\_name new\_name enclosure \_ | pathname}

Change the enclosure name of the enclosure or enclosures specified by the enclosure or pathname. The new name (\textit{new\_name}) must be 16 or less.
characters. Only alphabetic or numeric characters are acceptable. This subcommand applies only to the SENA.

**fc_s_download** [ -F ] [ -f fcode-file ]

Download the fcode contained in the file fcode-file into all the FC/S Sbus Cards. This command is interactive and expects user confirmation before downloading the fcode. When invoked without the -f fcode-file option, the current version of the fcode in each FC/S Sbus card is printed. When the -F option is used, the fcode is forcibly downloaded, but the command still expects user confirmation before the download. The version of the FC/S Sbus Cards fcode that was released with this version of the Operating System is kept in the directory `/usr/lib/firmware/fc_s` and is named `fc_s_fcode`.

Use **fc_s_download** only in single-user mode. Using **fc_s_download** to update a host adapter while there is I/O activity through that adapter will cause the adapter to reset.

**fcal_s_download** [ -f fcode-file ]

Download the fcode contained in the file fcode-file into all the FC100/S Sbus Cards. This command is interactive and expects user confirmation before downloading the fcode. When invoked without the -f option, the current version of the fcode in each FC100/S Sbus card is printed. The version of the FC100/S Sbus Cards fcode that was released with this version of the operating system is kept in the directory `/usr/lib/firmware/fc_s` and is named `fcal_s_fcode`.

Use **fcal_s_download** only in single-user mode. Using **fcal_s_download** to update a host adapter while there is I/O activity through that adapter will cause the adapter to reset.

**inquiry** enclosure[,dev] . . . | pathname . . .

Display the inquiry information for the selected device specified by the enclosure or pathname.

**insert_device** [ enclosure[,dev] . . . ] | pathname . . .

Assist the user in the hot insertion of a new device or a chain of new devices. Refer to **NOTES** for limitations on hotplug operations. This subcommand applies only to the SENA and the RSM subsystems. For the SENA, if more than one enclosure has been specified, concurrent hot insertions on multiple busses can be performed. With no arguments to the subcommand, entire enclosures can be inserted. For the RSM, only one controller can be specified. For the SENA, this subcommand guides the user interactively through the hot insertion steps of a new device or chain of devices. If a list of disks was entered it will ask the user to verify the list of devices to be inserted is correct, at which point the user can continue or quit. It then interactively asks the user to insert the disk(s) or enclosure(s) and then creates and displays the logical pathname(s) for the devices. For the RSM, the following steps are taken:

- Quiesce the bus or buses which support quiescing and
unquiescing.

- Inform the user that the device can be safely inserted.
- Request confirmation from the user that the device has been inserted.
- Unquiesce the bus or buses which support quiescing and unquiescing.
- Create the logical device name for the new device.

**led enclosure,dev... | pathname...**

Display the current state of the LED associated with the disk specified by the enclosure or pathname. This subcommand only applies to subsystems that support this functionality.

**led_blink enclosure,dev... | pathname...**

Requests the subsystem to start blinking the LED associated with the disk specified by the enclosure or pathname. This subcommand only applies to subsystems that support this functionality.

**led_off enclosure,dev... | pathname...**

Requests the subsystem to disable (turn off) the LED associated with the disk specified by the enclosure or pathname. On a SENA subsystem, this may or may not cause the LED to turn off or stop blinking depending on the state of the SENA subsystem. Refer to the SENA Array Installation and Service Manual (p/n 802-7573). This subcommand only applies to subsystems that support this functionality.

**led_on pathname...**

Requests the subsystem to enable (turn on) the LED associated with the disk specified by the enclosure or pathname. This subcommand only applies to subsystems that support this functionality.

**power_off enclosure[dev]... | pathname [enclosure-port]... | controller tray-number**

When a SENA is addressed, this subcommand causes the SENA subsystem to go into the power-save mode. The SENA drives are not available when in the power-save mode. When an Enclosure Services card within the SPARCstorage Array is addressed, the RSM tray is powered down. When a drive in a SENA is addressed the drive is set to the drive off/unmated state. In the drive off/unmated state, the drive is spun down (stopped) and in bypass mode.

**power_on enclosure[dev]... | pathname...**

Causes the SENA subsystem to go out of the power-save mode, when this subcommand is addressed to a SENA. There is no programmatic way to power on the SPARCstorage Array RSM tray. When this subcommand is addressed to a drive the drive is set to its normal start-up state.

**release pathname**

Release a reservation held on the specified disk. If the pathname is of the SPARCstorage Array controller, then all of the disks in the SPARCstorage...
Array will be released.

**remove_device enclosure[dev]... | pathname ...**
Assists the user in hot removing a device or a chain of devices. This subcommand can also be used to remove entire enclosures. This subcommand applies to the SENA and the RSM. Refer to **NOTES** for limitations on hotplug operations. For the SENA, this subcommand guides the user through the hot removal of a device or devices. During execution it will ask the user to verify the list of devices to be removed is correct, at which point the user can continue or quit. It then prepares the disk(s) or enclosure(s) for removal and interactively asks the user to remove the disk(s) or enclosure(s).

For the RSM, the steps taken are:
- Take the device offline.
- Quiesce the bus or buses which support quiescing and unquiescing.
- Inform user that the device can be safely removed.
- Request confirmation from the user that the device has been removed.
- Unquiesce the bus or buses which support quiescing and unquiescing.
- Bring the (now removed) device back online.
- Remove the logical device name for the removed device.

**replace_device pathname**
This subcommand applies only to the RSM. Refer to **NOTES** for limitations on hotplug operations. This subcommand guides the user interactively through the hot replacement of a device.

For the RSM, the steps taken are:
- Take the device offline.
- Quiesce the bus or buses which support quiescing and unquiescing.
- Inform user that the device can be safely replaced.
- Request confirmation from the user that the device has been replaced.
- Unquiesce the bus or buses which support quiescing and unquiescing.
- Bring the device back online.

**reserve pathname**
Reserve the specified disk for exclusive use by the issuing host. If the pathname is of the SPARCstorage Array controller, then all of the disks in the SPARCstorage Array will be reserved.

**set_boot_dev [ -y ] pathname**
Set the boot-device variable in the system PROM to the physical device name specified by *pathname*, which can be a block special device or a mount-point. The command normally runs interactively requesting
confirmation for setting the default boot-device in the PROM. The \(-y\) option can be used to run it non-interactively, in which case no confirmation is requested or required.

```
start \[-t \]\{tray-number\} pathame\ldots
```
Spin up the specified disk(s). If `pathname` specifies the SPARCstorage Array controller, this action applies to all disks in the SPARCstorage Array.
- `t` Spin up all disks in the tray specified by `tray-number`. `pathname` must specify the SPARCstorage Array controller.

```
stop \[-t \]\{tray-number\} pathame\ldots
```
Spin down the specified disk(s). If `pathname` specifies the SPARCstorage Array controller, this action applies to all disks in the SPARCstorage Array.
- `t` Spin down all disks in the tray specified by `tray-number`. `pathname` must specify the SPARCstorage Array controller.

**SPARCstorage Array Subcommands**

```
fast_write \[-s \]\[-c\] pathame
fast_write \[-s \]\[-d\] pathame
fast_write \[-s \]\[-e\] pathame
```
Enable or disable the use of the NVRAM to enhance the performance of writes in the SPARCstorage Array. `pathname` refers to the SPARCstorage Array controller or to an individual disk.
- `-s` Cause the SPARCstorage Array to save the change so it will persist across power-cycles.
- `-c` Enable fast writes for synchronous writes only.
- `-d` Disable fast writes.
- `-e` Enable fast writes.

```
vram_data pathame
```
Display the amount of fast write data in the NVRAM for the specified disk. This command can only be used for an individual disk.

```
perf_statistics \[-d \]\[-e\] pathame
```
Enable or disable the accumulation of performance statistics for the specified SPARCstorage Array controller. The accumulation of performance statistics must be enabled before using the display \(-psubcommand\). This subcommand can be issued only to the SPARCstorage Array controller.
- `-d` Disable the accumulation of performance statistics.
- `-e` Enable the accumulation of performance statistics.

```
purge pathame
```
Purge any fast write data from NVRAM for one disk, or all disks if the controller is specified. This option should be used with caution, usually only when a drive has failed.

```
sync_cache pathame
```
Flush all outstanding writes for the specified disk from NVRAM to the media. If `pathname` specifies the controller, this action applies to all disks in the SPARCstorage Array subsystem.

modified 20 Oct 1997

Solaris 2.6 Hardware: 3/98

1M-35
Enclosure Services
Card Subcommands

The env_display and alarm* subcommands apply only to an Enclosure Services Card (SES) in a RSM tray in a SPARCstorage Array. The RSM tray is addressed by using the logical or physical path of the SES device or by specifying the controller followed by the tray number. The controller is addressed by cN or the physical path to the SSA’s controller.

```plaintext
alarm pathname | controller tray_number
               Display the current state of audible alarm.
alarm_off pathname | controller tray_number
                    Disable the audible alarm for this RSM tray.
alarm_on pathname | controller tray_number
                    Enable the audible alarm for this RSM tray.
alarm_set controller-pathname | controller tray_number [ seconds ]
                               Set the audible alarm setting to seconds.
env_display pathname | controller tray_number
                      Display the environmental information for the specified unit.
```

SENA Expert Mode
Subcommands

The following subcommands are for expert use only, and are applicable only to the SENA subsystem. They should only be used by users that are knowledgeable about the SENA subsystem and fiber channel loops.

For the following subcommands that work on a bus if a disk is specified then the bus that disk attached to is used.

```plaintext
-e forcelip enclosure[dev] ... | pathname ...
Force the link to reinitialize, using the Loop Initialization Primitive (LIP) sequence. The enclosure or pathname can specify any device on the loop.
This is an expert only command and should be used with caution. It will reset all ports on the loop.
-e rdls enclosure[dev] ... | pathname ...
Read and display the link error status information for all available devices on the loop that contains the device specified by the enclosure or pathname.
```

Other Expert Mode
Subcommands

See NOTES for limitations of these subcommands. They should only be used by users that are knowledgeable about the systems they are managing.

```plaintext
-e bus_getstate pathname
   Get and display the state of the specified bus.
-e bus_quiesce pathname
   Quiesce the specified bus.
-e bus_reset pathname
   Reset the specified bus.
-e bus_unquiesce pathname
   Unquiesce the specified bus. the specified device.
```
- `e dev_getstate pathname`
  Get and display the state of the specified device.

- `e dev_reset pathname`
  Reset the specified device.

- `e offline pathname`
  Take the specified device offline.

- `e online pathname`
  Put the specified device online.

**EXAMPLES**

The following example finds and displays all of the SENAs on a system:

```bash
example% luxadm probe
```

The following example displays an SSA:

```bash
example% luxadm display c1
```

The following example displays a SENA:

```bash
example% luxadm display /dev/es/ses0
```

The following example displays of two subsystems using the enclosure names:

```bash
example% luxadm display BOB system1
```

The following example displays information about the first disk in the front of the enclosure named BOB. Use `f` to specify the front disks. Use `r` to specify the rear disks.

```bash
example% luxadm display BOB,
```

The following example displays information about a SENA disk or enclosure with the port WWN of `2200002037001246`:

```bash
example% luxadm display 2200002037001246
```

The following example uses only as many characters as are required to uniquely identify a subcommand:

```bash
example% luxadm disp BOB
```

The following example displays error information about the loop that the enclosure BOB is on:

```bash
example% luxadm display -r BOB
```

The following example downloads new firmware into the Interface Board in the enclosure named BOB (that this is using the default path for the file to download):

```bash
example% luxadm download -s BOB
```

The following example displays information from the SCSI inquiry command from all individual disks on the system, using only as many characters as necessary to uniquely identify the inquiry subcommand:

```bash
example% luxadm inq /dev/rdsk/c?t?d?s2
```

The following example hotplugs a new drive into the first slot in the front of the enclosure named BOB:

```bash
example% luxadm insert_device BOB,
```

modified 20 Oct 1997     Solaris 2.6 Hardware: 3/98     1M-37
The following example runs an expert subcommand. The subcommand forces a loop initialization on the loop that the enclosure BOB is on:

```
example % luxadm -e forcelip BOB
```

An example of using the expert mode hot plugging subcommands to hot remove a disk on a SSA follows. See NOTES for hot plugging limitations.

The first step reserves the SCSI device so that it can't be accessed by way of its second SCSI bus:

```
ex ample# luxadm reserve /dev/rdsk/c1t8d0s2
```

The next two steps take the disk to be removed offline then quiesce the bus:

```
ex ample# luxadm -e offline /dev/rdsk/c1t8d0s2
example# luxadm -e bus_quiesce /dev/rdsk/c1t8d0s2
```

The user then removes the disk and continues by unquiescing the bus, putting the disk back online, then unreserving it:

```
ex ample# luxadm -e bus_unquiesce /dev/rdsk/c1t8d0s2
example# luxadm -e online /dev/rdsk/c1t8d0s2
example# luxadm release /dev/rdsk/c1t8d0s2
```

ENVIRONMENT

See environ(5) for a description of the LANG environment variable that affects the execution of luxadm.

EXIT STATUS

The following exit values are returned:

0  Successful completion.

-1  An error occurred.

FILES

/usr/lib/firmware/fc/s/fcal_s_fcode
/usr/lib/firmware/fc/s/fc_s_fcode
/usr/lib/firmware/ssa/ssafirmware
/usr/lib/locale/C/LC_MESSAGES/ibfirmware

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

```
<table>
<thead>
<tr>
<th>ATTRIBUTE TYPE</th>
<th>ATTRIBUTE VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>SUNWluxop</td>
</tr>
</tbody>
</table>
```

SEE ALSO

devlinks(1M), disks(1M), ssaadm(1M), attributes(5), environ(5), ses(7D)


*SENA Array Installation and Service Manual* (p/n 802-7573).

NOTES

See the *SENA Array Installation and Service Manual* for additional information on the SENA. Refer to *Tutorial for SCSI use of IEEE Company_ID*, R. Snively, for additional information regarding the IEEE extended WWN. See SEE ALSO. Currently, only some device drivers support hot plugging. If hot plugging is attempted on a disk or bus where it is not supported, an error message of the form:

```
luxadm: can't acquire "PATHNAME": No such file or directory
```

will be displayed.
You must be careful not to quiesce a bus that contains the root or the /usr filesystems or any swap data. If you do quiesce such a bus a deadlock can result, requiring a system reboot.
NAME  
sm_configd – Solstice SyMON configuration reader

SYNOPSIS  
/opt/SUNWsymon/sbin/sm_configd [ -D debug-value ] [ -T file ] [ -i interval ]

AVAILABILITY  
SUNWsymon

DESCRIPTION  
Monitors the physical configuration of a machine and reports on the status of components. For further details, please see the Solstice SyMON User’s Guide.

OPTIONS  
- D  Set a debug option for ALL.
- T  Run the configuration from a file; for testing purposes.
- i  Set the polling interval for the Config Reader.

FILES  
cfg_sun4d.so.1

cfg_sun4u.so.1

cfg_sun4uI.so.1

SEE ALSO  
symon(1), sm_confsymon(1M), sm_control(1M), sm_egd(1M), sm_krd(1M),
sm_logscand(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4),
event_gen.tcl(4), logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)
NAME
sm_confsymon – configures the agent host and event monitor host machines running Solstice SyMON software

SYNOPSIS
sm_confsymon -s event_host [ -v ] [ -k polling_time ] [ -c polling_time ] [ -p ] [ -i sampling_time ] [ -U user_name ]
sm_confsymon -e server_host [ -M max_events ] [ -i sampling_time ] [ -S SNMP_hostname ] [ -P platform_name ] [ -d diskbox_name ]... [ -U user_name ]
sm_confsymon -D

AVAILABILITY
SUNWsymon

DESCRIPTION
sm_confsymon configures machines that are running Solstice SyMON software as an agent host (the server that is being monitored) and as the event monitor host (the machine that is monitoring the agent host).

This command is run on the respective machines used as agent host and event monitor host.

For further details on the operation of sm_confsymon please see the Solstice SyMON User’s Guide.

OPTIONS
- s Configures the server being monitored so it will identify the machine that is being used as the event monitor host. The machine name of monitoring machine is specified as event_host.
- v Selects verbose mode, in which the system will echo all actions performed.
- k Sets polling interval time for sm_krd to the number of seconds given as polling_time (default is 10 seconds).
- c Sets polling interval time for sm_confgd to the number of seconds given as polling_time (default is 10 seconds).
- p Modifies disk error message level in kernel and in /etc/system to log soft errors for PFA.
- i Sets sampling interval time to the number of seconds given as sampling_time (default is 10 seconds).
- U Sets the user ID used by sm_logscand (when included with the -s option) or sets the user ID used by sm_egd (when included with the -e option). The user ID is automatically generated when you provide the user name as the value of user_name.
- e Configures the machine doing the monitoring so it will identify the server that it is monitoring. The machine name of the monitored machine is specified as server_host.
- M Sets the maximum number of events, given as max_errors, before trimming (default is 1000 events).
- S Causes SNMP traps to be sent to the machine given as hostname.
Specifies the type of platform that is being monitored. This value, 
platform_name, is the result of running the `uname -i` command on the server 
being monitored (such as SUNW,SPARCserver-1000 ).
If you do not specify this option, `sm_symonconfig` will prompt you to enter 
the number of a platform type from a list it displays. Configuration will 
not continue until you specify the platform type. You can enter the number 
0 to exit at this point.

Specifies the type of disk storage box that is being monitored.

Completely removes the currently installed Solstice SyMON configuration.

SEE ALSO

`symon(1)`, `sm configd(1M)`, `sm control(1M)`, `sm egd(1M)`, `sm krd(1M)`, 
`sm_logscand(1M)`, `sm symond(1M)`, `auth_checker.tcl(4)`, `auth_list.tcl(4)`, 
`event_gen.tcl(4)`, `logscan.tcl(4)`, `rules.tcl(4)`, `sm_symond.conf(4)`
NAME  sm_control – starts or stops Solstice SyMON software on the server subsystem host or on the event generator machine.

SYNOPSIS  sm_control [ start ] [ stop ]

AVAILABILITY  SUNWsymon

DESCRIPTION  sm_control starts Solstice SyMON software on the server subsystem host machine or the event generator machine without needing to reboot the machine. It also can shut down the program on the machine. In either case, sm_control must be run as superuser on that machine.

For further details on the operation of sm_control please see the Solstice SyMON User's Guide.

OPTIONS

start  Starts Solstice SyMON software on a machine that has been configured as the server being monitored or the machine doing the monitoring.

stop  Shuts down the Solstice SyMON software.

SEE ALSO  symon(1), sm_configd(1M), sm_confsymon(1M), sm_egd(1M), sm_krd(1M), sm_logscand(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4), event_gen.tcl(4), logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)
NAME

sm_egd – Solstice SyMON event generator

SYNOPSIS


AVAILABILITY

SUNWsymon

DESCRIPTION

Monitors other symon agents and reports events based on Tcl rules defined in rules files.

OPTIONS

- Specify the polling interval (in seconds) when data is collected and rules are run.
- Specify a debug flag for the event generator. The following numbers can be added together to specify several debug options:
  1=Provides debugging on the initialization.
  2=Provides some basic Tcl debugging.
  4=Provides debuggin information on basic calls to rules and AIL.
  8=Provides data on the rules as understood by the event generator.
  16=Provides debugging on AIL callbacks.
  32=Provides debugging on building match lists for MULTI rules.
  64=Provides debugging on agent births and deaths.
- Specify the location of the event generator log file.
- Specify a file used by the event generator to track event numbers.
- Specify a rules file. This file must contain the Rules variable in Tcl.
- Specify a file to initialize Tcl procedures.
- Specify a shared object to be loaded. This option must be used in conjunction with the -f option.
- Specifies the function within a shared object that will be called when this object is loaded. This option must be used in conjunction with the -l option.
- Specifies the name of the root for the outgoing hierarchy.
- Specifies an AIL debugging flag. The following numbers can be added together to specify several AIL debug options:
  1=Print AIP version.
  2=List of hierarchy updates.
  4=Trace requests and connections.
  8=Tell if replacing an existing node.
  16=Debug pruning.
32=Trace memory use.
64=Report sm_symond traffic.
128=Sleep 30 seconds before starting.
256=Fake server death if /tmp/dead exists.
512=Print out strings used.
1024=Print messages showing time for AIP transactions.

- B  Specifies the directory for storing the event database.
- t  Specifies the target machine to be polled.
- S  Specifies that core dumps are allowed.
- P  Specifies that process data should be polled.
- L  Specifies the location of a Tcl library.
- U  Specifies a user name under which to run the event generator program.

Specifies an RPC number for connecting to
     sm_symond.

- V  Specifies a directory for running the event generator. (This can override the
     location set by the -t option. However, the -h, -H, or -B flag can override
     the location specified in the -V flag.)

FILES

rules.tcl  Specifies the rules, in Tcl, for the event generator. Located in
            /etc/opt/SUNWsymon.

event_gen.tcl  The initialization file for the event generator. Located in
               /etc/opt/SUNWsymon.

event_log  The log file for events. Located in /var/opt/SUNWsymon/target.

EG_events  Stores the last event number. Located in
            /var/opt/SUNWsymon/target.

events/*/  Each event in the all events hierarchy. Located in
            /var/opt/SUNWsymon/target.

SEE ALSO

symon(1), sm_configd(1M), sm_confsymon(1M), sm_control(1M), sm_krd(1M),
sm_logscand(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4),
event_gen.tcl(4), logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)
NAME  

sm_krd – Solstice SyMON kernel reader

SYNOPSIS  

/opt/SUNWsymon/sbin/sm_krd [ -d ] [ -DAIL-debug-flag ] [ -v ]  
[ -t ] [ -r ] [ -R ] [ -U kernel-file ] [ -M kmem-file ] [ -S swap-file ]  
[ -i interval ] [ -P count ] [ -T count ]

AVAILABILITY  

SUNWsymon

DESCRIPTION  

sm_krd monitors the kernel on an active machine, and reports data to clients. For more information, please see the Solstice SyMON User's Guide.

OPTIONS  

- **d**  
  Activate Kernel Reader debugging.

- **D**  
  Specify an AIL debugging level (values can be added together for combinations of debug output):
  1=print AIP version  
  2=list of hierarchy updates  
  4=trace requests and connections  
  8=tell if replacing an existing node  
  10=debug pruning  
  20=trace memory use  
  40=report sm_symond traffic  
  80=sleep 30 seconds before starting  
  100=fake server death if /tmp/dead exists

- **v**  
  Run the kernel reader in verbose mode.

- **t**  
  Set the timer flag.

- **r**  
  Set the resource information flag.

- **R**  
  Set the resource information summary flag.

- **U**  
  Specify the name of the kernel file.

- **M**  
  Specify the name for the kmem file.

- **S**  
  Specify the name of the swap file.

- **i**  
  Specify the polling interval.

- **P**  
  Run for the specified number of intervals, then quit.

- **T**  
  Build the tree for debugging.

- **count**  
  Automatically report data for every count intervals.

SEE ALSO  

symon(1), sm_configd(1M), sm_confsymon(1M), sm_control(1M), sm_egd(1M),  
sm_logscand(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4),  
event_gen.tcl(4), logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)
**NAME**  
sm_logscand – Solstice SyMON log file scanner

**SYNOPSIS**  
/opt/SUNWsymb/sbin/sm_logscand [ -i interval ] [ -L TCL-library ] [ -U user-name ] log-definition-file

**AVAILABILITY**  
SUNWsymb

**DESCRIPTION**  
Scans the log files, as described in the log definition file.

**OPTIONS**  
- Set the polling interval to update log files.
- Specify the location of the Tcl library.
- Specify a user name for running the program.

**FILES**  
log-definition-file Initialization file for the log scanner. Located in /etc/opt/SUNWsymon.

**SEE ALSO**  
symon(1), sm_configd(1M), sm_configsymb(1M), sm_control(1M), sm_eld(1M), sm_krd(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4), event_gen.tcl(4), logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)
NAME

sm_symond – Solstice SyMON process controller

SYNOPSIS

/opt/SUNWsymon/sbin/sm_symond [ -n RPC-number ]
[ -d debug-level ] [ -D AIL-debug-level ] [ -p output-level ]
[ -P minutes ] [ -i intervals ] [ -A file ] [ -C file ]
[ -E directory ] [ -H directory ] [ -I directory ] [ -L file ]

AVAILABILITY

SUNWsymon

DESCRIPTION

sm_symond is a tool to manage Solstice SyMON processes. Its primary role is to start
the program’s agents, monitor those agents for crashes, and provide RPC information
to clients that wish to access any of those agents.

The primary repository for agent data is the file
/etc/opt/SUNWsymon/sm_symond.conf (see sm_symond.conf(4)).

When sm_symond is run, it first reads /etc/opt/SUNWsymon/sm_symond.conf to
determine the local agents to be spawned. It then spawns those agents. If an entry
indicates that an agent may exist on a remote system, sm_symond will poll that system
looking for another symond to get information on that agent.

Symond serves a hierarchy of information via RPC to any requesting client. Each
agent should produce a hierarchy that is readable.

sm_symond is also responsible for looking at the auth_checker.tcl and auth_list.tcl
scripts to determine if a Solstice SyMON user has access to the symon data.

OPTIONS

- n Specify a custom RPC number for this program (the default is 100244). If you
use this option to specify a different number for the monitored host, you must
also supply it to any client programs, such as symon or sm_egd. This option
does not dissociate process and child agents.

- d Debugging level for sm_symond. These values can be added together for com-
binations of debug output:
  1=trace
  2=callbacks
  4=rpc
  8=spawn info
  16=debug access control
  32=config file info

- D Debugging level for AIL for hierarchy transport.

- p Print hierarchy level:
  1=nodes
  5=nodes and prop
  10=nodes, prop, and data

- P Turn on profiling to dump after specified number of minutes.

- i Sampling interval for checking if the agents are still alive.
-A  Specifies alternative authorization checking file (default is auth_checker.tcl).
-C  Specifies alternative configuration file (default is sm_symond.conf).
-E  Specifies an alternative “etc” directory (default is /etc/opt/SUNWsymon).
-H  Specifies an alternative “home” directory (default is /var/opt/SUNWsymon).
   sm_symond will run from inside a subdirectory called hostname under this directory. Any core file or debug file that is generated will reside there.
-I  Specifies an alternative install directory (default is /opt/SUNWsymon). This contains a subdirectory called etc containing authorization files that are used if no authorization files are found in the directory specified by the –E option. This also contains a subdirectory called lib/tcl that contains the Tcl library.
-L  Specifies an alternative authorization list file (default is auth_list.tcl).

FILES
   /etc/opt/SUNWsymon/sm_symond.conf

SEE ALSO
   symon(1), sm_configd(1M), sm_confsymon(1M), sm_control(1M), sm_egd(1M),
   sm_krd(1M), sm_logscand(1M), auth_checker.tcl(4), auth_list.tcl(4), event_gen.tcl(4),
   logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)

NOTES
   sm_symond can only be run by root.
NAME

ftpd – FTP daemon that runs on the Sun MediaCenter. Enables use of standard ftp commands for moving content.

SYNOPSIS

ftp [-dgintv] [hostname]

AVAILABILITY

Available with the Sun MediaCenter server software. On a Sun MediaCenter server, this binary replaces the ftpd that is shipped with Solaris.

DESCRIPTION

ftpd is the FTP daemon shipped with the Sun MediaCenter server. It supports all standard ftp commands, plus commands (of the same names as standard commands) that support the movement of video content between a local file system and the Media File System (MFS) on a Sun MediaCenter server. This means that you can load content onto a Sun MediaCenter server from any platform that has an FTP-protocol-conformant ftp client.

Note: The FTP daemon described here is a superset of the standard FTP daemon. Thus, this man page supplements the ftpd (1M) man page that is shipped with Solaris.

The video-file functions of the FTP daemon are invoked with the keyword smc: For example, you enter a command such as the following to obtain a listing of all titles on a Sun MediaCenter server:

    ftp> ls smc:title=*  

Note that you must use a backslash (\) to escape the asterisk.

OPTIONS

See the ftp (1) man page for a description of that program's options. The Sun MediaCenter FTP daemon supports all of the standard ftp options, on all platforms.

VIDEO FILE ATTRIBUTES

Video content is stored on the Sun MediaCenter server in data and index files that collectively make up a title. A title is identified by a string of the format:

    smc:attr_name=value,[attr_name=value] ...

A fully-qualified title identifier has the form:

    smc:name=name,speed=speed,type=[data|index],rate=rate,format=format

Title attributes are described as follows:

name

Name of the movie or video clip. There is no default value.

speed

Refers to the speed and play direction of the title's bit stream, as compared to normal-play, forward direction. The default is 1000, meaning normal play speed, forward direction.
type

"Data" or "index". A data file contains an MPEG bit stream. An index file identifies splice points within a bit stream. The default is "data".

rate

Rate at which the file containing the video bit stream was encoded, expressed in bits per second. Applies only to data files, not index files. There is no default value.

format

Format of the bit stream. Can be either MPEGTS or MPEG1SYS. Other stream formats are supported by the server, but not by the FTP daemon. Note that for MPEGTS-format titles, the FTP daemon automatically generates index files for titles that contain trick play streams.

FTP COMMANDS

Listed below are the ftp commands for video files supported by the Sun MediaCenter FTP daemon. These commands accept video file attributes as arguments. Some use only a name; others require a name plus other attributes.

FTP allows the use of the asterisk (*) wildcard character in specifying filenames. You must use a backslash (\) to escape the asterisk. (Some PC-based implementations of FTP clients do not require a backslash.) For video files, the asterisk stands for "all video files," including both data and index files.

The video-file commands have the same semantics as the Solaris implementation of the ftp commands.

The example commands assume that the user has successfully established an FTP connection with a Sun MediaCenter server.

You should always use binary mode when transferring video files.

delete

ftp> del smc:title=title_name

dir

ftp> dir smc:title=title_name

get

ftp> get smc:title=title_name,speed=speed,type=[data|index] path_to_local_file

For get, you should specify, in addition to name, the speed and type attributes. If you do not specify speed and type, they take default values, which might not be appropriate for your title. The rate and format attributes are recommended, especially if you might later need to put files back on a Sun MediaCenter server. For example, if you use get to backup titles, specify rate and format so that, if you ever need to restore titles (using put), the values for those attributes will be available.
SMC_FTPD (1M)

Maintenance Commands

ls

    ftp> ls smc:title=title_name

mget

    ftp> mget smc:title=\* path_to_local_file

mls

    ftp> mls smc:title=title_name [smc:title=title_name]... output_file

For output file, you can use a hyphen (-) to indicate stdout.

mput

    ftp> mput local_files

For mput, local_files must have the same format as used for the destination argument for put. See the following command.

put

    ftp> put local_file smc:title=title_name, speed=speed,\n        type=[data|index], rate=rate, format=format

If you do not specify speed and type attributes for put, the default values are used. You must specify the rate and format attributes for this command.

rename

    ftp> rename smc:title=title_name

rename accepts only a name attribute. You are prompted for a new name after entering the command.

EXAMPLES

All examples assume a successful FTP connection with a Sun MediaCenter server.

    ftp> dir smc:
    ftp> ls smc:title=\* 

The two preceding commands return a list of the titles stored on the server, with their attributes.

    ftp> get smc:title=bambi,speed=1000,type=data,rate=3000000 \ 
        /home/backup/bambi.data

The preceding command copies the title "Bambi", with relevant attributes, to a file in the local file system.

    ftp> put /home/backup/batman.data \ 
        smc:title=bambi,speed=1000,type=data,rate=3000000,format=MPEG1SYS

The preceding command copies the data file for "Bambi" from a local file system to a Sun MediaCenter server. The following sequence might be used to backup and restore video files on a Sun MediaCenter server:
1. Establish FTP connection to Sun MediaCenter server:
   
   # ftp server_name

   Logon as root.

2. Check on titles:

   ftp> ls smc:title=\* smc:title=bambi,format=MPEGTS,speed=1000,type=data,rate=3072000

3. Use output from previous command to backup titles:

   ftp> get smc:title=bambi,format=MPEGTS,speed=1000,type=data,rate=3072000 /home/backup/bambi.vid

   You might also use: mget smc:title=\*

   ftp> put /home/backup/bambi.vid /smc:title=bambi,format=MPEGTS,rate=3072000

   In the preceding command, note that the speed and type attributes are not specified. Speed defaults to 1000 and type defaults to data, which are appropriate choices for this example. Also note that format and rate are specified, which is a requirement for a put command.

SEE ALSO

The Sun MediaCenter Administrator’s Guide

smc_copy (1), smc_tar (1), smc_ls (1), smc_rm (1)
NAME  
smc_gettacl – obtain access control list for titles on Sun MediaCenter server

SYNOPSIS  
smc_gettacl [ server: ] <title name>...

AVAILABILITY  
Available with the Sun MediaCenter Server software. smc_gettacl is a companion command to smc_settacl (1M)

DESCRIPTION  
smc_gettacl allows you to obtain the access control list (ACL) associated with a title on a Sun MediaCenter server. Output from smc_gettacl is suitable as input for the -f option of smc_settacl. It is useful to pipe output from smc_gettacl to smc_settacl to set the ACL for a title to be the same as another title’s ACL.

OPTIONS  
smc_gettacl has no options. It accepts as an argument:
[ server: ] <title name>...
You can specify one or more titles, any of which can be local or remote. Specify multiple title names with a space between each pair. For a remote title, you prepend the name of the Sun MediaCenter server and a colon to the title name. You can use an asterisk in the <title name> field, which means all titles on the server. You must use a backslash (\) to escape the asterisk.

EXAMPLES  
The following command obtains the ACL for the local title "bambi" and the remote title "ben_hur", which is stored on the server "nicene".

% smc_gettacl bambi nicene:ben_hur

The following command pipes output from smc_gettacl to smc_settacl, setting the ACL for "bambi" to match that of "ben_hur".

% smc_gettacl nicene:ben_hur | smc_settacl -f - bambi

SEE ALSO  
smc_tar (1), smc_copy (1), smc_settacl (1M)
NAME

`smc_settacl` – set title access control list for Sun MediaCenter server

SYNOPSIS

```
smc_settacl -s <acl_entries> [ server: ] <titlename>...
smc_settacl -m <acl_entries> [ server: ] <titlename>...
smc_settacl -d <title_users> [ server: ] <titlename>...
smc_settacl -f <filename> [ server: ] <titlename>...
```

AVAILABILITY

Available with the Sun MediaCenter Server software.

DESCRIPTION

`smc_settacl` allows you to set, modify, or delete the access control list (ACL) associated with a title on a Sun MediaCenter server. After copying a video file (title) to a server, you must use `smc_settacl` if you want other users to be able to copy, append to, or delete that title.

OPTIONS

`-s <acl_entries> [ server: ] <titlename>...`

Replace the current title ACL with an ACL containing the information specified in `<acl_entries>`. `<acl_entries>` stands for a comma-separated list of items of the form:

```
u[ser]:<username>:<permissions>
```

`<username>` is a Solaris login name; `<permissions>` is one or more of r, w, and a (read, write, and admin, respectively). You specify permissions in the order rwa. Replace any permission you are not setting with a hyphen. So, for example, if you are setting only admin permission, you specify `-a`; if you are setting only read and admin, specify `r-a`. Permissions are defined in the `Sun MediaCenter Server Programmer's Guide`.

`-m <acl_entries> [ server: ] <titlename>...`

Modify the current title ACL according to `<acl_entries>`. If you specify a user who is not in the title ACL, that user is appended to the ACL. If you specify a user who is in the ACL, the permissions for that user are changed to what you specify.

`-d <title_users> [ server: ] <titlename>...`

From the ACL for a specified title, deletes users specified in `<title_users>`, which is a comma-separated list of items of the form:

```
u[ser]: <username>
```

where `<username>` is a Solaris login name.

`-f <filename> [ server: ] <titlename>...`

Set the ACL(s) for the specified title(s) according to the contents of `<filename>`, a text file containing a list of entries of the form of `<acl_entries>`, above, with one entry per line. You can have comments in the file; comments are indicated by a hash mark in column 1.

You cannot use the `-s` and `-f` options with any other option. You can combine `-m` and `-d`.

modified 14 April 1997

Solaris 2.6 Hardware: 3/98
For the `-s`, `-m`, and `-d` options and in an entry in a file introduced by `-f`, you can use an asterisk in the user field, which means "any user". For all options, you can specify one or more titles, any of which can be local or remote. Specify multiple title names with a space between each pair. For a remote title, you prepend the name of the Sun MediaCenter server and a colon to the title name. You can use an asterisk in the title name field to stand for all titles on a server. You must use a backslash (`\`) to escape the asterisk.

**EXAMPLES**

The following command replaces an ACL associated with the title "bambi" with an ACL that allows the user "srinivasan" read and admin access.

```
% smc_settacl -s u:srinivasan:r a bambi
```

The following command modifies the ACL associated with the title "bambi", adding the user "srinivasan", with read and admin access.

```
% smc_settacl -m u:srinivasan:r a bambi
```

The following command deletes the user "srinivasan" from the ACL for the title "bambi" on the remote server "nicene".

```
% smc_settacl -d u:srinivasan nicene:bambi
```

The following command sets the ACLs for all titles on the remote server "nicene" according to the contents of the file "acl_list".

```
% smc_settacl -f /home/admin/acl_list nicene:\*
```

**SEE ALSO**

`smc_tar (1), smc_copy (1), smc_getacl (1M)`
NAME
ssp-config – set initial SSP configuration information on the host

DESCRIPTION Caution: Never execute this command on the command line.

/ssr/platform/sbin/bin/ssp-config is normally invoked by the /etc/init.d/sspdefs startup
script during boot of the Enterprise 10000 host, but only if the file ./SSP_DEFAULTS
exists. **ssp-config** interactively prompts for information, including the SSP’s hostname
and IP address. It uses the information to set the initial configuration to allow com-
munication between the server and the SSP.

Only super user can run **ssp-config**.

FILES
./SSP_DEFAULTS
/etc/inet/hosts
/etc/ssphostname
/etc/syslog.conf

SEE ALSO
ssp-unconfig(1M)
NAME
ssp-unconfig – undo SSP and system information on the host

DESCRIPTION
Caution: Only super user can use this command. Exercise extreme caution in its use.

When executed on an Enterprise 10000 server /usr/platform/sun4u1/sbin/ssp-unconfig removes configuration information established by the command ssp-config(1M), then invokes the SunOS command sys-unconfig(1M) to make the system ready to be configured again.

The ssp-unconfig command does the following:

- Removes SSP information from the /etc/syslog.conf and /etc/inet/hosts files.
- Removes the /etc/ssphostname file.

When finished, ssp-unconfig, invokes the SunOS command sys-unconfig(1M), which performs a system shutdown.

FILES
/.SSP_DEFAULTS
/etc/inet/hosts
/etc/ssphostname
/etc/syslog.conf

SEE ALSO
ssp-config(1M) in this reference manual

sys-unconfig(1M) in man Pages(1M): System Administration Commands
NAME
sunvts – Invokes the SunVTS kernel and its user interface

SYNOPSIS
sunvts [ -lepqstv ] [ -o option_file ] [ -f log_dir ] [ -h hostname ]

AVAILABILITY
SUNWvts

DESCRIPTION
The sunvts command is used to invoke the SunVTS user interface and kernel on the
same system. It could be used to start the user interface on the local system and con-
nect to the SunVTS kernel on the remote system. By default, it displays CDE Motif
graphic interface for CDE environment, OpenLook graphic interface for OpenWindows
environment, or TTY interface for non-windowing system.

OPTIONS
-1 Displays SunVTS OpenLook graphic interface.
-e Disables the security checking feature.
-f log_dir
   Specifies an alternative log_file directory. The default log_file directory is
   /var/opt/SUNWvts/logs.
-h hostname
   Starts the SunVTS user interface on the local system, which connects to or
   invokes the SunVTS kernel on the specified host after security checking
   succeeds.
-o option_file
   Starts the SunVTS kernel with the test options loaded from the specified
   option_file, which by default is located in /var/opt/SUNWvts/options.
-p Starts the SunVTS kernel vtsk (1M) such that it does not probe the test
   system’s devices.
-q Automatically quits both the SunVTS kernel and the user interface when test-
   ing stops.
-s Automatically starts testing from a selected group of tests. The flag must be
   used with the -o option_file flag.
-t Starts vtstty (1M), a TTY based interface, instead of CDE or OpenLook inter-
   face.
-v Displays version information from vtsui(1M) and vtsk(1M).

NOTES
If vtsk (1M) is already running on the test system, the sunvts command ignores the
-e, -o, -f, -q, -p, and -s options.

SEE ALSO
vtsk(1M), vtstty(1M), vtsui(1M), vtsui.ol(1M), vtsprobe(1M)

modified 15 Mar 1996
NAME  
vtsk – SunVTS diagnostic kernel

SYNOPSIS  
vtsk [-epqsv] [-o options_file] [-f logfile_directory]

AVAILABILITY  
SUNWvts

DESCRIPTION  
The `vtsk` command starts up the SunVTS diagnostic kernel as a background process. There can only be one copy of `vtsk` running at a time. Only the superuser can execute this command.

Normally, `vtsk` is automatically started up by the `sunvts(1M)` command if it is not already running. `vtsk` will also be invoked by `inetd(1M)` when there is a connection request from `vtsui` or `vtsui.ol`. In that case, the security file, `.sunvts_sec`, will be checked for the permission before running vtsk on the target host specified by `vtsui(1M)` or `vtsui.ol(1M)`.

OPTIONS  
- `e` Enables the security checking for all connection requests.
- `p` Starts SunVTS diagnostic kernel, but does not probe system configuration.
- `q` Quits both the SunVTS diagnostic kernel and the attached User Interfaces when the testing is completed.
- `s` Runs enabled tests immediately after started.
- `v` Display SunVTS diagnostic kernel’s version information only.
- `-o options_file` Starts the SunVTS diagnostic kernel and sets the test options according to the option file named `options_file`.
- `-f logfile_directory` Specifies an alternative logfile directory, other than the default.

EXIT STATUS  
The following exit values are returned:
0  Successful completion.
-1  An error occurred.

FILES  
/var/opt/SUNWvts/options  default option file directory.
/var/opt/SUNWvts/logs  default log file directory.

SEE ALSO  
sunvts(1M), vtsui(1M), vtsui.ol(1M), vtstty(1M), vtsprobe(1M)
**NAME**  
`vtsprobe` – prints the device probe information from the SunVTS kernel

**SYNOPSIS**  
`vtsprobe [-m] [-h hostname]`

**AVAILABILITY**  
SUNWvts

**DESCRIPTION**  
`vtsprobe` is a utility that displays the device and configuration information contained in the SunVTS kernel. The output includes the SunVTS assigned group for the device, the device name, the device instance, the testname attached to this device, and the configuration information obtained from the device-specific test probe.

**OPTIONS**  
- `-m` Specifies manufacturing mode, which displays the probe information in a format that is easy to read using script files.
- `-h hostname`  
  Specifies the `hostname` to connect to and get the device and configuration information. If not specified, the current host will be used.

**USAGE**  
After the SunVTS kernel is up and running, you may type `vtsprobe` at the shell prompt to get the probe output. (See the `sunvts (1M)` man page for more information on how to start up SunVTS.

**EXAMPLE**  
Running `vtsprobe` on a sun4m SPARCclassic produces the following output:

```
% vtsprobe

Processor(s)
  system(systest)
    System Configuration=sun4m SPARCclassic
    System clock frequency=50 MHz
    SBUS clock frequency=25 MHz
  fpu(fputest)
    Architecture=sparc
    Type=TI TMS390S10 or TMS390S15 microSPARC chip

Memory
  kmem(vmem)
    Total: 143120KB
  mem(pmem)
    Physical Memory size=24 Mb

SCSI-Devices(esp0)
  c0l2d0(rawtest)
    Capacity: 638.35MB
    Controller: esp0
    Vendor: MICROP
    SUN Id: 1588-15MBSUN0669
    Firmware Rev: SN0C
```

modified 15 Mar 1996

Solaris 2.6 Hardware: 3/98

1M-61
Serial Number: 1588-15MB103
c0t2d0(fstest)
    Controller: esp0
c0t3d0(rawtest)
    Capacity: 404.65MB
    Controller: esp0
    Vendor: SEAGATE
    SUN Id: ST1480    SUN0424
    Firmware Rev: 8628
    Serial Number: 00836508
c0t3d0(fstest)
    Capacity: 404.65MB
    Controller: esp0
    Vendor: SEAGATE
    SUN Id: ST1480    SUN0424
    Firmware Rev: 8628
    Serial Number: 00836508
c0t3d0(fstest)
    Controller: esp0
c0t6d0(cdtest)
    Controller: esp0
tape1(tapetest)
    Drive Type: Exabyte EXB-8500 8mm Helical Scan
Network
    isdn0(isdntest)
        NT Port    TE Port
le0(nettest)
    Host_Name: ctech84
    Host Address: 129.146.210.84
    Host ID: 8001784b
    Domain Name: scsict.Eng.Sun.COM
Comm.Ports
    zs0(sptest)
        Port a -- zs0    /dev/term/a : /devices/ ... a
        Port b -- zs1    /dev/term/b : /devices/ ... b
Graphics
cgthree0(fbtest)
OtherDevices
    bpp0(bpptest)
        Logical name: bpp0
    sound0(audio)
        Audio Device Type: AMD79C30
    sound1(audio)
        Audio Device Type: DBRI Speakerbox

Solaris 2.6 Hardware: 3/98  modified 15 Mar 1996
spd0(spdtest)
Logical name: spd0

NOTES
The output of vtsprobe is highly dependent on the device being correctly configured into the system (so that a SunVTS probe for the device can be run successfully on it) and on the availability of a device-specific test probe. If the device is improperly configured or if there is no probing function associated with this device, vtsprobe cannot print any information associated with it.

SEE ALSO sunvts(1M), vtsk(1M), vtsui(1M), vtsui.ol(1M), vtstty(1M)
NAME
vtstty – TTY interface for SunVTS

SYNOPSIS
vtstty [ -q ] [ -h hostname ]

AVAILABILITY
SUNWvts

DESCRIPTION
vtstty is the default interface for SunVTS in the absence of a windowing environment. It can be used in a non-windowing environment such as a terminal connected to the serial port of the system. However, its use is not restricted to this; vtstty can also be used from shell window.

OPTIONS
- q The "auto-quit" option automatically quits when the conditions for SunVTS to quit are met.
- v Prints the vtstty version. The interface is not started when you include this option.
- h hostname
    Connects to the SunVTS kernel running on the host identified by hostname.

USAGE
The vtstty screen consists of four panels: main control, status, test groups, and console. The panels are used to display choices that the user can select to perform some function and/or to display information. A panel is said to be "in focus" or in a "selected" state when it is surrounded by asterisks and the current item is highlighted. In order to choose from the items in a panel, the focus should be shifted to that panel first.

The following are the different types of selection items that can be present in a panel:

Text string
Describes a choice that, when selected, either pops up another panel or performs a function. For example, "stop" will stop the SunVTS testing.

Data entry field
To enter or edit numeric or textual data.

Checkbox
Represented as "[ ]". Checkboxes are associated with items and indicate whether the associated item is selected or not. A checkbox can be in one of the following two states: Deselected [ ] or Selected [*].

The key assignments given below describe the keys for shifting focus, making a selection, and performing other functions:

TAB or <CTRL>W Shift focus to another panel
RETURN Select current item
Spacebar Toggle checkbox
Up arrow or <CTRL>U Move up one item
Down arrow or <CTRL>N Move down one item
Left arrow or <CTRL>P
  Move left one item
Right arrow or <CTRL>R
  Move right one item
Backspace
  Delete text in a data entry field
ESC
  Dismiss a pop-up
<CTRL>F
  Scroll forward in a scrollable panel
<CTRL>B
  Scroll backward in a scrollable panel
<CTRL>X
  Quit vtstty but leave the SunVTS kernel running
<CTRL>L
  Refresh the vtstty screen

NOTES
1. To run vtstty from a telnet session, carry out the following steps:
   a. Before telnet-ing, determine the values for "rows and "columns". (See stty(1)).
   b. Set term to the appropriate type after telnet-ing(for example, set term=vt100
   c. Set the values of columns and rows to the value noted above. (See stty(1)).
2. Before running vtstty ensure that the environment variable describing the terminal type is set correctly.

SEE ALSO sunvts(1M), vtsk(1M), vtsui(1M), vtsui.ol(1M), vtsprobe(1M)
NAME
vtsui – SunVTS Graphic User Interface (CDE)

SYNOPSIS
vtsui [ -qv ] [ -h hostname ]

AVAILABILITY
SUNWvts

DESCRIPTION
The vtsui command starts up the CDE Motif version of SunVTS graphic user interface. There can be multiple instances of vtsui running at the same time, all connected to one SunVTS diagnostic kernel, vtsk(1M). The name of the host machine running the diagnostic kernel, vtsk(1M), will be displayed in the title bar of the graphical user interface window.

vtsui is automatically started up by the sunvts (1M) command. vtsui can be also used to start vtsk (1M) if inetd (1M) is in operation. In that case, the security file, sunvts_sec, will be checked for the permission before running vtsk on the target host.

See the "SunVTS User’s Guide" for a complete description on using the graphical user interface.

OPTIONS
- q    Quits the SunVTS graphic user interface when testing has terminated.
- v    Displays graphic user interface version information only.
- h hostname    Starts the SunVTS graphic user interface and connects to the SunVTS diagnostic kernel running on hostname, or invokes the kernel if not running, after security checking succeeds. If hostname not specified, the local host is assumed.

EXIT STATUS
The following exit values are returned:
0    Successful completion.
1    An error occurred.

SEE ALSO
sunvts(1M), vtsk(1M), vtsui.ol(1M), vtstty(1M), vtsprobe(1M)
NAME  vtsui.ol – SunVTS Graphic User Interface (OpenLook)

SYNOPSIS  vtsui.ol [ -qv ] [ -h hostname ]

AVAILABILITY  SUNWvts

DESCRIPTION  The vtsui.ol command starts up the OpenLook version of SunVTS graphic user interface. There can be multiple instances of vtsui.ol running at the same time, all connected to one SunVTS diagnostic kernel, vtsk(1M). The name of the host machine running the diagnostic kernel, vtsk(1M), will be displayed in the title bar of the graphic user interface window.

vtsui.ol can be used to start vtsk(1M) if inetd(1M) is in operation. In that case, the security file, .sunvts_sec, will be checked for the permission before running vtsk on the target host. vtsui.ol is also automatically started up by the sunvts(1M) command. See the "SunVTS User's Guide" for a complete description on using the graphic user interface.

OPTIONS  

  -q  Quits the SunVTS graphic user interface when testing has terminated.
  -v  Displays graphic user interface version information only.
  -h hostname  Starts the SunVTS graphic user interface and connects to the SunVTS diagnostic kernel running on hostname, or invokes the kernel if not running, after security checking succeeds. If hostname not specified, the local host is assumed.

EXIT STATUS  The following exit values are returned:

  0  Successful completion.
  1  An error occurred.

SEE ALSO  sunvts(1M), vtsk(1M), vtsui(1M), vtstty(1M), vtsprobe(1M)
NAME
config_change_state, etc. – Configuration administration interface

SYNOPSIS
cc [flag ...] file ... -lcfgadm -ldl [ library.... ]
#include <config_admin.h>

cfga_err_t config_change_state(cfga_cmd_t state_change_cmd, int num_ap_ids, char *
const *ap_ids,
    const char *options, struct cfga_confirm *confp, struct cfga_msg *msgp, char
**errstring,
    cfga_flags_t flags);

cfga_err_t config_private_func(const char *function, int num_ap_ids, char * const
*ap_ids,
    const char *options, struct cfga_confirm *confp, struct cfga_msg *msgp, char
**errstring,
    cfga_flags_t flags);

cfga_err_t config_test(int num_ap_ids, char * const *ap_ids, const char *options,
    struct cfga_msg *msgp, char **errstring cfga_flags_t flags);

cfga_err_t config_stat(int num_ap_ids, char * const *ap_ids, struct cfga_stat_data *buf,
    const char *options char **errstring);

cfga_err_t config_list(struct cfga_stat_data **ap_id_list, int *nlist, const char *options
char **errstring);

cfga_err_t config_ap_id_cmp(const cfga_ap_id_t ap_id1, const cfga_ap_id_t ap_id2);
void config_unload_libs();
const char *config_strerror(cfga_err_t cfgerrnum);

HARDWARE
DEPENDENT
LIBRARY
SYNOPSIS

cfga_err_t config_change_state(cfga_cmd_t state_change_cmd, const char *ap_id, const
char *options,
    struct cfga_confirm *confp, struct cfga_msg *msgp, char **errstring, cfga_flags_t
flags);

cfga_err_t config_private_func(const char *function, const char *ap_id, const char
*options,
    struct cfga_confirm *confp, struct cfga_msg *msgp, char **errstring, cfga_flags_t
flags);

cfga_err_t config_test(const char *ap_id, const char *options, struct cfga_msg *msgp, char
**errstring,
    cfga_flags_t flags);

cfga_err_t config_stat(const char *ap_id, struct cfga_stat_data *buf, const char *options
char **errstring);

cfga_err_t config_list(struct cfga_stat_data **ap_id_list, int *nlist, const char *options
char **errstring);
The `confgxxxx` routines provide a hardware independent interface to hardware
specific system configuration administration functions. The `cfgaxxxx` routines are pro-
vided by hardware specific libraries that are dynamically loaded to handle
configuration administration functions in a hardware specific manner.

The `libcfgadm` library is used to provide the services of the `cfgadm(1M)` command.
The `libcfgadm` hardware specific libraries are located in `/usr/lib/cfgadm`. The
hardware specific library names are obtained from properties in device tree nodes that
identify attachment points, or derived from the caller supplied `ap_id`.

The `confg change state` routine performs functions that change the state of the system
configuration. The `state change cmd` can be one of the following: `CFGA CMD_PREP`,
`CFGA CMD LOAD`, `CFGA CMD UNLOAD`, `CFGA CMD RESUME`,
`CFGA CMD DISCONNECT`, `CFGA CMD_CONNECT`, `CFGA CMD_CONFIGURE`
or `CFGA CMD UNCONFIGURE`. The `state change cmd CFGA CMD_PREP` is used
to prepare a receptacle for physical addition or removal of an occupant. The
`state change cmd CFGA CMD LOAD` is used to activate automatic hardware insertion
of an occupant. The `state change cmd CFGA CMD_UNLOAD` is used to activate
automatic hardware removal of an occupant. The `state change cmd
CFGA CMD_RESUME` is used to exit the `CFGA STAT_PREP` state after the phy-
sical addition or removal of an occupant. The `state change cmd
CFGA CMD_DISCONNECT` is used to disable normal communication to or from an
occupant in a receptacle. The `state change cmd CFGA CMD_CONNECT` is used to
enable communication to or from an occupant in a receptacle. The `state change cmd
CFGA CMD_CONFIGURE` is used to bring the hardware resources contained on, or
attached to, an occupant into the realm of Solaris, allowing use of the occupant’s
hardware resources by the system. The `state change cmd
CFGA CMD_UNCONFIGURE` is used to remove the hardware resources contained
on, or attached to, an occupant from the realm of Solaris, disallowing further use of
the occupant’s hardware resources by the system.

The `flags argument may contain one or both of the defined flags,
`CFGA_FLAG_FORCE` and `CFGA_FLAG_VERBOSE`. If the `CFGA_FLAG_FORCE`
flag is asserted certain safety checks will be overridden. For example, this may not
allow an occupant in the failed condition to be configured, but might allow an occu-
pant in the failing condition to be configured. Acceptance of a force is hardware
dependent. If the `CFGA_FLAG_VERBOSE` flag is asserted hardware specific details
relating to the operation are output utilizing the `cfga_msg` mechanism.
The \texttt{\texttt{config\_private\_func}} routine invokes non-state changing private hardware specific functions.

The \texttt{config\_test} routine is used to initiate testing of the specified attachment point.

The \texttt{num\_ap\_ids} argument specifies the number of \texttt{ap\_ids} in the \texttt{ap\_ids} array. The \texttt{ap\_ids} argument points to an array of \texttt{ap\_ids}.

The \texttt{ap\_id} argument points to a single \texttt{ap\_id}.

The \texttt{function} and \texttt{options} strings conform to the \texttt{getsubopt(3C)} syntax convention and are used to supply hardware specific function or option information. No generic hardware independent functions or options are defined.

The \texttt{cfga\_confirm} structure referenced by \texttt{confp} provides a call-back interface to get permission to proceed should the requested operation require, for example, a noticeable service interruption. The \texttt{cfga\_confirm} structure includes the following members:

\begin{verbatim}
  int  (*confirm)(void *appdata\_ptr, const char *message);
  void *appdata\_ptr;
\end{verbatim}

The \texttt{confirm} function is called with two arguments: The generic pointer \texttt{appdata\_ptr} and the message detailing what requires confirmation. The generic pointer \texttt{appdata\_ptr} is set to the value passed in in the \texttt{cfga\_confirm} structure member \texttt{appdata\_ptr} and can be used in a graphical user interface to relate the \texttt{confirm} function call to the \texttt{config\_xxxx} call. The \texttt{confirm} function should return one (1) to allow the operation to proceed and zero (0) otherwise.

The \texttt{cfga\_msg} structure referenced by \texttt{msgp} provides a call-back interface to output intermediate messages from a hardware specific library. In the presence of the \texttt{CFG\_FLAG\_VERBOSE} flag these messages can be informational, otherwise they are restricted to error messages. The \texttt{cfga\_msg} structure includes the following members:

\begin{verbatim}
  void (*message\_routine)(void *appdata\_ptr, const char *message);
  void *appdata\_ptr;
\end{verbatim}

The \texttt{message\_routine} function is called with two arguments: The generic pointer \texttt{appdata\_ptr} and the message. The generic pointer \texttt{appdata\_ptr} is set to the value passed in in the \texttt{cfga\_confirm} structure member \texttt{appdata\_ptr} and can be used in a graphical user interface to relate the \texttt{message\_routine} function call to the \texttt{config\_xxxx} call. The messages will be in the native language specified by the \texttt{LC\_MESSAGES} locale category; see \texttt{setlocale(3C)}.

For some generic errors a hardware specific error message can be returned. The storage for the error message string, including the terminating null character, is allocated by the \texttt{config\_xxxx} functions using \texttt{malloc(3C)} and a pointer to this storage returned through \texttt{errstring}. If \texttt{errstring} is \texttt{NULL} no error message will be generated or returned. If \texttt{errstring} is not \texttt{NULL} and no error message is generated, the pointer referenced by \texttt{errstring} will be set to \texttt{NULL}. It is the responsibility of the function calling \texttt{config\_xxxx} to deallocate the returned storage using \texttt{free(3C)}. The error messages will be in the native language specified by the \texttt{LC\_MESSAGES} locale category; see \texttt{setlocale(3C)}.
The `config_stat` routine provides a way of getting status for an attachment point. The `cfga_stat_data` structure includes the following members:

```c
typedef char cfga_ap_id_t[CFGAP_ID_LEN];
typedef char cfga_info_t[CFGA_INFO_LEN];
typedef char cfga_type_t[CFGA_TYPE_LEN];
typedef enum cfga_stat_t;
typedef enum cfga_cond_t;
typedef enum cfga_activity_t;
typedef int cfga_flags_t;
```

The `ap_log_id` and the `ap_phys_id` fields give the hardware specific logical and physical names of the attachment point. The `ap_activity` field indicates activity that may result in changes to state or condition. The `ap_status_time` field gives the time at which either the `ap_r_state`, `ap_o_state` or `ap_cond` fields of the attachment point, last changed. The field `ap_info` provides additional information, such as speed, that may be available from the hardware dependent code.

The fields `ap_log_id`, `ap_phys_id`, `cfga_info_t` and `cfga_type_t` are null terminated strings. When printing these fields the following format is suggested:

```c
printf("%.s", sizeof(p->ap_log_id), p->ap_log_id);
```

The `config_list` routine provides a way of obtaining the status of all attachment points in the system. The function returns an array of `cfga_stat_data` structures, one for each attachment point in the system. The storage for the array is allocated by the `config_list` function using `malloc(3C)` and a pointer to this storage returned through `ap_id_list`. The number of array elements is returned through `nlist`. It is the responsibility of the function calling `config_list` to deallocate the returned storage using `free(3C)`.

The `config_ap_id_cmp` function performs a hardware dependent comparison on two `ap_ids`, returning an equal to, less than or greater than indication in the manner of `strcmp(3C)`. Each argument is either a `cfga_ap_id_t` or can be a null terminated string. This function can be used when sorting lists of `ap_ids`, for example with `qsort(3C)`, or when selecting entries from the result of a `config_list` function call.

The `config_unload_libs` function unlinks all previously loaded hardware specific libraries.

modified 5 Nov 1997

Solaris 2.6 Hardware: 3/98
The `config_strerror` function can be used to map an error return value (see below) to an error message string. The returned string should not be overwritten.

`config_strerror` returns NULL if `cfgerrnum` is out-of-range.

**RETURN VALUES**

The list below gives the possible values returned by `config_xx` and `cfga_xx` routines. Additional error information may be returned through `errstring`, as defined above, if the return code is not `CFGA_OK`.

- **CFGA_OK**
  - The command completed as requested.

- **CFGA_NACK**
  - The command was not completed due to a negative acknowledgement from the `confp->confirm` function.

- **CFGA_NOTSUPP**
  - System configuration administration is not supported on the specified target.

- **CFGA_OPNOTSUPP**
  - System configuration administration operation is not supported on this attachment point.

- **CFGA_PRIV**
  - The caller does not have the required process privileges. For example, if configuration administration is performed through a device driver, the permissions on the device node would be used to control access.

- **CFGA_BUSY**
  - The command was not completed due to an element of the system configuration administration system being busy.

- **CFGA_SYSTEM_BUSY**
  - The command required a service interruption and was not completed due to a part of the system that could not be quiesced.

- **CFGA_LIB_ERROR**
  - A procedural error occurred in the library, including failure to obtain process resources such as memory and file descriptors.

- **CFGA_NO_LIB**
  - A hardware specific library could not be located using the supplied `ap_id`.

- **CFGA_ERROR**
  - An error occurred during the processing of the requested operation. This error code includes validation of the command arguments by the hardware specific code.

**ERRORS**

Many of the errors returned by the system configuration administration functions are hardware specific. The strings returned in `errstring` may include the following:

- **attachment point `ap_id` not known**
  - The attachment point detailed in the error message does not exist.

- **unknown hardware option `option` for operation**
An unknown option was encountered in the options string.

**hardware option option requires a value**
An option in the options string should have been of the form `option=value`.

**hardware option option does not require a value**
An option in the options string should have been a simple option.

**attachment point ap_id is not configured**
A config_change_state command to CFGA_CMD_UNCONFIGURE an occupant was made to an attachment point whose occupant was not in the CFGA_STAT_CONFIGURED state.

**attachment point ap_id is not unconfigured**
A config_change_state command requiring an unconfigured occupant was made to an attachment point whose occupant was not in the CFGA_STAT_UNCONFIGURED state.

**attachment point ap_id condition not satisfactory.**
A config_change_state command was made to an attachment point whose condition prevented the operation.

**attachment point ap_id in condition condition cannot be used**
A config_change_state operation with force indicated was directed to an attachment point whose condition fails the hardware dependent test.

SEE ALSO cfgadm(1M), dlopen(3x), dlsym(3x).

NOTES
Applications using this library should be aware that the underlying implementation may use system services which alter the contents of the external variable `errno` and may use file descriptor resources.

The following code shows the intended error processing when config_change_state, config_private_func, config_test, config_stat or config_list returns a value other than CFGA_OK:

```c
void emit_error(int cfgerrnum, char *estrp)
{
    const char *ep;

    ep = config_strerror(cfgerrnum);
    if (ep == NULL)
        ep = gettext("configuration administration unknown error");
    if (estrp != NULL && *estrp != '0') {
        (void) fprintf(stderr, "%s: %s\n", ep, estrp);
    } else {
        (void) fprintf(stderr, "%s\n", ep);
    }
    if (estrp != NULL)
        free((void *)estrp);
}
```

modified 5 Nov 1997 Solaris 2.6 Hardware: 3/98
Reference should be made to the Hardware Specific Guide for details of System Configuration Administration support.
NAME
auth_checker.tcl – Parser for handling list of authorized Solstice SyMON users

SYNOPSIS
/opt/SUNWsymon/etc/auth_checker.tcl

DESCRIPTION
This Tcl file parses the list of authorized Solstice SyMON users contained in the auth_list.tcl(4) file.
For more information, see the Solstice SyMON User’s Guide

SEE ALSO
symon(1), sm_configd(1M), sm_confsymon(1M), sm_control(1M), sm_egd(1M),
sm_krd(1M), sm_logscand(1M), sm_symond(1M), auth_list.tcl(4), event_gen.tcl(4),
logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)
NAME  auth_list.tcl – List of authorized Solstice SyMON users

SYNOPSIS  /opt/SUNWsymon/etc/auth_list.tcl

DESCRIPTION  This list identifies the users authorized to use the Solstice SyMON software on a system. Users, hosts, and groups can be defined as authorized, readonly, or unauthorized.

The data in auth_list.tcl is parsed by auth_checker.tcl(4).

For more information, see the Solstice SyMON User’s Guide

SEE ALSO  symon(1), sm_configd(1M), sm_confsymon(1M), sm_control(1M), sm_egd(1M), sm_krd(1M), sm_logscand(1M), sm_symond(1M), auth_checker.tcl(4), event_gen.tcl(4), logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)
NAME  event_gen.tcl – Defines procedures and variables used by rules in the Solstice SyMON program

SYNOPSIS  /opt/SUNWsymon/etc/event_gen.tcl

DESCRIPTION  When you run the `sm_confsymon -e servername` command, the `event_gen.tcl` file is copied to create a file called `event_gen.servername.tcl` that contains information specific to that machine within the Solstice SyMON program.

This information includes the host names of machines that will be sent snmp trap messages.

For more information, see the Solstice SyMON User’s Guide.

SEE ALSO  symon(1), sm_configd(1M), sm_confsymon(1M), sm_control(1M), sm_egd(1M), sm_krd(1M), sm_logsand(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4), logscan.tcl(4), rules.tcl(4), sm_symond.conf(4)

modified 2 Nov 1996  Solaris 2.6 Hardware: 3/98  4-77
NAME logscan.tcl – Defines file that the Solstice SyMON program’s Log Viewer will search

SYNOPSIS /opt/SUNWsymon/etc/logscan.tcl

DESCRIPTION This Tcl file contains a definition of the /var/adm/messages file that will be searched by the Log Viewer of the Solstice SyMON program.
For more information, see the Solstice SyMON User’s Guide

SEE ALSO symon(1), sm_confld(1M), sm_confsymon(1M), sm_control(1M), sm_eld(1M),
sm_krd(1M), sm_logscand(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4),
event_gen.tcl(4), rules.tcl(4), sm_symond.conf(4)
NAME    rules.tcl – The master set of event rules used by Tcl software in the Solstice SyMON program

SYNOPSIS /opt/SUNWsymon/etc/rules.tcl

DESCRIPTION This Tcl file contains a master list of event rules. When you create a new rules file, add a psource command for the new rules file to the rules.tcl file so that the new rules file can be read. For more information, see the Solstice SyMON User's Guide

SEE ALSO symon(1), sm_configd(1M), sm_confsymon(1M), sm_control(1M), sm_egd(1M), sm_krd(1M), sm_logscand(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4), event_gen.tcl(4), logscan.tcl(4), sm_symond.conf(4)
NAME
sm_symond.conf – list of agents for sm_symond to spawn and retrieve from other hosts

DESCRIPTION
The file /etc/opt/SUNWsymon/sm_symond.conf controls process spawning by sm_symond(1M). The processes most typically dispatched by sm_symond are symon agents.

The sm_symond.conf file is composed of entries that either list an agent and its arguments, or specify agents to run on remote machines.

Local agents are listed, one per line, with the normal command line arguments, and are invoked by sm_symond. Remote agent entries have the following format:

host agent-type

Each entry is delimited by a newline. Comments may be inserted in the sm_symond.conf file by starting the line with a #.

The remote agent fields are:

host
The name of the remote host where the agent is to be run.

agent-type
The specific type of symon agent being run. Currently, the only agent type supported on remote machines is EventGenerator.

SEE ALSO
symon(1), sm_configd(1M), sm_confsymon(1M), sm_control(1M), sm_egd(1M), sm_krd(1M), sm_logscand(1M), sm_symond(1M), auth_checker.tcl(4), auth_list.tcl(4), event_gen.tcl(4), logscan.tcl(4), rules.tcl(4)
NAME  afb – Elite3D graphics accelerator driver

DESCRIPTION  afb is the device driver for the Sun Elite3D graphics accelerators.
The afbdaemon process loads the afb microcode at system startup time and during the
resume sequence of a suspend-resume cycle.

FILES  
/dev/fbs/afb  device special file
/usr/lib/afb.ucode  afb microcode
/usr/sbin/afbdaemon  the afb microcode loader

SEE ALSO  afbconfig(1M),
### NAME
cvc – virtual console driver

### DESCRIPTION
cvc is a STREAMS-based pseudodriver that supports the network console, which is called cvc on the host side and netcon on the SSP. cvc interfaces with console(7). Logically, the cvc driver sits below the console(7) driver. It intercepts console output, redirecting it to the cvcredir(7) driver. cvc receives console input from cvcredir(7) and passes it to the process associated with /dev/console.

### NOTES
The cvc facility supercedes the SunOS wscons(7) facility, which should **not** be used in conjunction with cvc. wscons(7) is useful for systems with directly attached consoles (frame buffers and keyboards), but is not useful with the Enterprise 10000 system, which has no local keyboard or frame buffer.

### SEE ALSO
cvcd(1M), cvc(7), cvcredir(7) in this reference manual
netcon(1M), netcon_server(1M) in *UNKNOWN TITLE ABBREVIATION: UE10000REFEMAN1M*
console(7) in man Pages(7): Device and Network Interfaces
NAME
cvcredir – virtual console redirection driver

DESCRIPTION
cvcredir, the virtual console redirection driver, is a STREAMS-based pseudodriver that works in conjunction with the cvc driver, cvc(7), and the cvc daemon, cvcd(1M).

The cvcredir device is opened at start-of-day by the cvc daemon, cvcd(1M). cvcredir receives console output from cvc(7) and passes it to cvcd(1M). It receives console input from cvcd(1M) and passes it to cvc(7).

SEE ALSO
cvcd(1M), cvc(7) in this reference manual

netcon(1M), netcon_server(1M) in man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands

console(7) in man Pages(7): Device and Network Interfaces
NAME

dr – dynamic reconfiguration driver, /dev/dr

SYNOPSIS

dr

DESCRIPTION

The DR driver provides a pseudo-driver interface to the kernel Dynamic
Reconfiguration (DR) Attach and DR Detach features.

For DR Detach, the command `dr_daemon(1M)` executes SunOS `ioctl(2)` calls to:

- Detach selected devices from kernel usage
- Remove detached device nodes from the kernel’s device tree
- Direct OBP to delete all detached nodes from its device tree

For DR Attach, `dr_daemon(1M)` executes `ioctl(2)` calls to:

- Direct OBP to probe the board and add nodes to its device tree
- Get the nodes from OBP and add proto nodes to the kernel’s device tree
- Convert the proto nodes to CF1 nodes

The pathname of the device node is `/devices/pseudo/dr@0:0`.

SEE ALSO

Dynamic Reconfiguration User’s Guide
Ultra Enterprise 10000 SSP User’s Guide

`dr_daemon(1M)` in this reference manual

`hostview(1M), hpost(1M)` in `man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands`

`dr(1M)` in `man Pages(1M): DR Administration Commands`

`add_drv(1M), drvconfig(1M), devlinks(1M), disks(1M), ports(1M), tapes(1M)` in `man Pages(1M): System Administration Commands`
NAME  idn – Inter-Domain Network device driver

DESCRIPTION  idn is a multi-thread, loadable, clonable, STREAMS-based pseudo driver that supports the connectionless Data Link Provider Interface, dlpi(7P), over the Enterprise 10000 Gigplane-XB Interconnect. This connection is permitted only between domains within the same Enterprise 10000 server.

The driver provides one to 32 logical network interfaces. One or more of these interfaces may be connected to one or more dynamic system domains that have been previously linked to the local domain via the domain_link(1M) command. (See domain_link(1M) in the Ultra Enterprise 10000 SSP 3.1 Reference.) The driver works in conjunction with the SSP to perform domain linking and unlinking, along with automated linking upon host bootup.

The cloning character-special device /dev/idn is used to access all IDN services provided by the system.

idn and DLPI  The idn driver is a "style 2" Data Link Service provider. All M_PROTO and M_PCPROTO type messages are interpreted as DLPI primitives. An explicit DL_ATTACH_REQ message by the user is required for idn to associate the opened stream with a particular device (ppa). The ppa ID is interpreted as an unsigned long and indicates the corresponding device instance (unit) number. The error DL_ERROR_ACK is returned by the driver if the ppa field value does not correspond to a valid device instance number for the system. The device is initialized on first attach and de-initialized (stopped) on last detach.

The values returned by the driver in the DL_INFO_ACK primitive in response to the DL_INFO_REQ from the user are as follows:

- The maximum SDU is configurable via ndd(1M) and has the range of 512 bytes to 512K bytes. The default value is 16384 bytes.
- The minimum SDU is 0.
- The dlsap address length is 8.
- The MAC type is DL_ETHER.
- The sap length value is -2, meaning the physical address component is followed immediately by a 2-byte sap component within the DLSAP address.
- The service mode is DL_CLDLS.
- No optional quality of service (QOS) support is included at present so the QOS fields are 0.
- The provider style is DL_STYLE2.
- The version is DL_VERSION_2.
- The broadcast address value is Ethernet/IEEE broadcast address (0xFFFFFFFF). Note that IDN supports broadcast by issuing messages to each target individually. IDN is inherently a point-to-point network between domains. Once in the DL_ATTACHED state, the user must send a DL_BIND_REQ to associate a particular
SAP (Service Access Pointer) with the stream. The idn driver interprets the sap field within the DL_BIND_REQ as an Ethernet “type” therefore valid values for the sap field are in the [0-0xFFFF] range. Only one Ethernet type can be bound to the stream at any time.

If the user selects a sap with a value of 0, the receiver will be in 802.3 mode. All frames received from the media having a “type” field in the range [0-1500] are assumed to be 802.3 frames and are routed up all open Streams which are bound to sap value 0. If more than one Stream is in “802.3 mode” then the frame will be duplicated and routed up multiple Streams as DL_UNITDATA_IND messages.

In transmission, the driver checks the sap field of the DL_BIND_REQ if the sap value is 0, and if the destination type field is in the range [0-1500]. If either is true, the driver computes the length of the message, not including initial M_PROTO mblk (message block), of all subsequent DL_UNITDATA_REQ messages and transmits 802.3 frames that have this value in the MAC frame header length field.

The driver also supports raw M_DATA mode. When the user sends a DLIOCRAW ioctl, the particular Stream is put in raw mode. A complete frame along with a proper ether header is expected as part of the data.

The idn driver DLSAP address format consists of the 6 byte physical (Ethernet) address component followed immediately by the 2 byte sap (type) component producing an 8-byte DLSAP address. Applications should not hardcode to this particular implementation-specific DLSAP address format but use information returned in the DL_INFO_ACK primitive to compose and decompose DLSAP addresses. The sap length, full DLSAP length, and sap/physical ordering are included within the DL_INFO_ACK. The physical address length can be computed by subtracting the sap length from the full DLSAP address length or by issuing the DL_PHYS_ADDR_REQ to obtain the current physical address associated with the stream.

Once in the DL_BOUND state, the user may transmit frames on the IDN by sending DL_UNITDATA_REQ messages to the idn driver. The idn driver will route received IDN frames up all those open and bound streams having a sap which matches the Ethernet type as DL_UNITDATA_IND messages. Received IDN frames are duplicated and routed up multiple open streams if necessary. The DLSAP address contained within the DL_UNITDATA_REQ and DL_UNITDATA_IND messages consists of both the sap (type) and physical (Ethernet) components.

**idn Primitives**

In addition to the mandatory connectionless DLPI message set the driver additionally supports the following primitives.

The DL_ENABMULTI_REQ and DL_DISABMULTI_REQ primitives enable/disable reception of individual multicast group addresses. A set of multicast addresses may be iteratively created and modified on a per-stream basis using these primitives. These primitives are accepted by the driver in any state following DL_ATTACHED.

The DL_PROMISCON_REQ and DL_PROMISCOFF_REQ primitives with the DL_PROMISC_PHYS flag set in the dl_level field enables/disables reception of all (“promiscuous mode”) frames on the media including frames generated by the local
domain. When used with the DL_PROMISC_SAP flag set this enables/disables reception of all sap (Ethernet type) values. When used with the DL_PROMISC_MULTI flag set this enables/disables reception of all multicast group addresses. The effect of each is always on a per-stream basis and independent of the other sap and physical level configurations on this stream or other streams.

The DL_PHYS_ADDR_REQ primitive return the 6 octet Ethernet address currently associated (attached) to the stream in the DL_PHYS_ADDR_ACK primitive. This primitive is valid only in states following a successful DL_ATTACH_REQ.

The DL_SET_PHYS_ADDR_REQ primitive is not allowed by the idn driver as the driver maintains point-to-point domain address information in the address in order to direct packets to the correct destination.

NOTES

The driver supports a set of tuneable parameters. The list can be retrieved via ndd(1M).

FILES

/dev/idn idn special character device.

SEE ALSO

Inter-Domain Network User's Guide
domain_link(1M), domain_unlink(1M) in the Ultra Enterprise 10000 SSP Reference Manual


modified January 1998
Index

A
afb (7D), 7-81
afbconfig (1M), 1M-10
auth_checker.tcl (4), 4-756
auth_list.tcl (4), 4-76

C
cfgadm (1M), 1M-18
cfg_admin (3X), 3-68
cvc (7), 7-82
cvcd (1M), 1M-27
cvcredir (7), 7-83

D
devices and network interfaces, 7-81 to 7-87
dr (7), 7-84
dr_daemon (1M), 1M-28

E
event_gen.tcl (4), 4-77

F
file formats, 4-75 to 4-80

I
idn (7), 7-85

L
library functions, miscellaneous, 3-68 to 3-74
logscan.tcl (4), 4-78
luxadm (1M), 1M-29

M
maintenance commands, 1M-10 to 1M-67

R
rules.tcl (4), 4-79

S
sm_configd (1M), 1M-40
sm_confsymon (1M), 1M-41
sm_control (1M), 1M-43
sm_egd (1M), 1M-43
sm_krd (1M), 1M-46
sm_logscand (1M), 1M-47
sm_symond (1M), 1M-48
sm_symond.conf (4), 4-41
smc_copy (1), 1-1
smc_ftpd (1M), 1M-50
smc_gettacl (1M), 1M-54
smc_1s (1), 1-3
smc_rm (1), 1-4
smc_settacl (1M), 1M-55
smc_tar (1), 1-5
ssp-config (1M), 1M-57
ssp-unconfig (1M), 1M-58
sunvts (1M), 1M-59
symon (1), 1-7

U
user commands, 1-1 to 1-9

V
vtsk (1M), 1M-60
vtsprobe (1M), 1M-61
vttstty (1M), 1M-64
vtsui (1M), 1M-66
vtsui.ol (1M), 1M-67