NAME

sg_ses_microcode − send microcode to a SCSI enclosure

SYNOPSIS

sg_ses_microcode [*−−bpw=CS*] [*−−dry−run*] [*−−ealsd*] [*−−help*] [*−−id=ID*] [*−−in=FILE*] [*−−length=LEN*] [*−−mode=MO*] [*−−non*] [*−−offset=OFF*] [*−−skip=SKIP*] [*−−subenc=MS*] [*−−tlength=TLEN*] [*−−verbose*] [*−−version*] *DEVICE*

DESCRIPTION

This utility attempts to download microcode to an enclosure (or one of its sub−enclosures) associated with the *DEVICE*. The process for doing this is defined in the SCSI Enclosure Services (SES) standards and drafts maintained by the T10 committee.

The process is to send one or more sequences containing a SCSI SEND DIAGNOSTIC command followed optionally by a RECEIVE DIAGNOSTIC RESULTS command. The former sends a Download microcode Control diagnostic page (dpage) and the latter fetches a Download microcode status dpage which can be viewed as a report on the former command.

The default action (i.e. when the *−−mode=MO* option is not given) is to fetch the Download microcode status dpage and decode it. This does not require the microcode (firmware) itself so the *−−in=FILE* option is not required.

The most recent reference for this utility is the draft SCSI Enclosure Services 3 (SES−3) document T10/2149−D Revision 7 at http://www.t10.org . Existing standards for SES and SES−2 are ANSI INCITS 305−1998 and ANSI INCITS 448−2008 respectively.

Most other support for SES in this package (apart from downloading microcode) can be found in the sg_ses utility. Another way of downloading firmware to a SCSI device is with the WRITE BUFFER command defined in SPC−4, see the sg_write_buffer utility.

OPTIONS

Arguments to long options are mandatory for short options as well.

−b, **−−bpw**=*CS*

where *CS* is the chunk size in bytes and should be a multiple of 4. This will be the maximum number of bytes sent per SEND DIAGNOSTIC command. So if *CS* is less than the effective length of the microcode then multiple SEND DIAGNOSTIC commands are sent, each taking the next chunk from the read data and increasing the buffer offset field in the Download microcode control dpage by the appropriate amount. The default is a chunk size of 0 which is interpreted as a very large number hence only one SEND DIAGNOSTIC command will be sent.
The number in *CS* can optionally be followed by ",act" or ",activate". In this case after the microcode has been successfully sent to the *DEVICE*, an additional Download microcode control dpage with its mode set to "Activate deferred microcode" [0xf] is sent.

−d, **−−dry−run**

the actual calls to perform SEND DIAGNOSTIC and RECEIVE DIAGNOSTIC RESULTS commands are skipped when this option is given. No SCSI commands are sent to the *DEVICE* but it is still opened and is required to be given. A dummy device such as /dev/null (in Unix) can be used.
This utility expects a "sensible" response to the RECEIVE DIAGNOSTIC RESULTS command it sends (and will abort if it doesn't receive one). So this option supplies dummy responses with one primary enclosure and three sub−enclosures. The dummy responses include good status values.

−e, **−−ealsd**

exit after last SEND DIAGNOSTIC command. A SES device should not start its firmware update immediately after the last received "chunk" of its firmware. Rather it should wait till at least one RECEIVE DIAGNOSTIC RESULTS command is sent to give the device a chance to report any error. However some devices do start the firmware update immediately which causes the trailing RECEIVE DIAGNOSTIC RESULTS command to be held up and often be aborted with a "target reset" error.
This option causes the trailing RECEIVE DIAGNOSTIC RESULTS command to be skipped. This

option would be typically used with the *−−bpw=CS* option.
Prior to version 1.10 of this utility [20180112] this (i.e. skipping the last RECEIVE DIAGNOSTIC RESULTS command) was the default action.

−h, **−−help**
output the usage message then exit. If used multiple times also prints the mode names and their acronyms.

−i, **−−id**=*ID*
this option sets the BUFFER ID field in the Download microcode control dpage. *ID* is a value between 0 (default) and 255 inclusive.

−I, **−−in**=*FILE*
read data from file *FILE* that will be sent with the SEND DIAGNOSTIC command. If *FILE* is '−' then stdin is read until an EOF is detected (this is the same action as *−−raw*). Data is read from the beginning of *FILE* except in the case when it is a regular file and the *−−skip=SKIP* option is given.

−l, **−−length**=*LEN*
where *LEN* is the length, in bytes, of data to be written to the device. If not given (and the length cannot be deduced from *−−in=FILE* or *−−raw*) then defaults to zero. If the option is given and the length deduced from *−−in=FILE* or *−−raw* is less (or no data is provided), then bytes of 0xff are used as fill bytes.

−m, **−−mode**=*MO*
this option sets the MODE. *MO* is a value between 0 (which is dmc_status and the default) and 255 inclusive. Alternatively an abbreviation can be given. See the MODES section below. To list the available mode abbreviations at run time give an invalid one (e.g. '−−mode=xxx') or use the '−h' option.

−N, **−−non**
allow for non−standard implementations that reset their Download microcode engine after a RECEIVE DIAGNOSTIC RESULTS command with the Download microcode status dpage is sent. When this option is given sending that command and dpage combination is avoided unless an error has already occurred.

−o, **−−offset**=*OFF*
this option sets the BUFFER OFFSET field in the Download microcode control dpage. *OFF* is a value between 0 (default) and 2**32−1 . It is a byte offset. This option is ignored (and a warning sent to stderr) if the *−−bpw=CS* option is also given.

−s, **−−skip**=*SKIP*
this option is only active when *−−in=FILE* is given and *FILE* is a regular file, rather than stdin. Data is read starting at byte offset *SKIP* to the end of file (or the amount given by *−−length=LEN*). If not given the byte offset defaults to 0 (i.e. the start of the file).

−S, **−−subenc**=*SEID*
SEID is the sub−enclosure identify. It defaults to 0 which is the primary enclosure identifier.

−t, **−−tlength**=*TLEN*
TLEN is the total length in bytes of the microcode to be (or being) downloaded. It defaults to 0 which is okay in most cases. This option only comes into play when *TLEN* is greater than *LEN*. In this case *TLEN* is sent to the SES *DEVICE* so that it knows when it only receives *LEN* bytes from this invocation, that it should expect more to be sent in the near future (e.g. by another invocation). This option is only needed when sections of microcode are being sent in separate invocations of this utility (e.g. the microcode is spread across two files).

−v, **−−verbose**
increase the level of verbosity, (i.e. debug output).

−V, **−−version**
print the version string and then exit.

MODES

Following is a list accepted by the *MO* argument of this utility. First shown is an acronym followed in square brackets by the corresponding decimal and hex values that may also be given for *MO*.

dmc_status [0, 0x0]
Use RECEIVE DIAGNOSTIC RESULTS to fetch the Download microcode status dpage and print it out.

dmc_offs [6, 0x6]
Download microcode with offsets and activate.

dmc_offs_save [7, 0x7]
Download microcode with offsets, save, and activate.

dmc_offs_defer [14, 0xe]
Download microcode with offsets, save, and defer activate.

activate_mc [15, 0xf]
Activate deferred microcode. There is no follow−up RECEIVE DIAGNOSTIC RESULTS to fetch the Download microcode status dpage since the *DEVICE* might be resetting.

Apart from dmc_status, these are placed in the Download microcode mode field in the Download microcode control dpage. In the case of dmc_status the Download microcode status dpage is fetched with the RECEIVE DIAGNOSTIC RESULTS command and decoded.

WHEN THE DOWNLOAD FAILS

Firstly, if it succeeds, this utility should stay silent and return. Typically vendors will change the "revision" string (which is 4 characters long) whenever they release new firmware. That can be seen in the response to a SCSI INQUIRY command, for example by using the sg_inq utility. It is possible that the device needs to be power cycled before the new microcode becomes active. Also if mode dmc_offs_defer [0xe] is used to download the microcode, then another invocation with activate_mc may be needed.

If something goes wrong, there will typically be messages printed out by this utility. The first thing to check is the microcode (firmware) file itself. Is it designed for the device model; has it been corrupted, and if downgrading (i.e. trying to reinstate older firmware), does the vendor allow that?

Getting new firmware on a device is a delicate operation that is not always well defined by T10's standards and drafts. One might speculate that they are deliberately vague. In testing this utility one vendor's interpretation of the standard was somewhat surprising. The *−−non* option was added to cope with their interpretation. So if the above suggestions don't help, try adding the *−−non* option.

NOTES

This utility can handle a maximum size of 128 MB of microcode which should be sufficient for most purposes. In a system that is memory constrained, such large allocations of memory may fail.

The user should be aware that most operating systems have limits on the amount of data that can be sent with one SCSI command. In Linux this depends on the pass through mechanism used (e.g. block SG_IO or the sg driver) and various setting in sysfs in the Linux lk 2.6/3 series (e.g. /sys/block/sda/queue/max_sectors_kb). Devices (i.e. logical units) also typically have limits on the maximum amount of data they can handle in one command. These two limitations suggest that modes containing the word "offset" together with the *−−bpw=CS* option are required as firmware files get larger and larger. And *CS* can be quite small, for example 4096 bytes, resulting in many SEND DIAGNOSTIC commands being sent.

The exact error from the non−standard implementation was a sense key of ILLEGAL REQUEST and an asc/ascq code of 0x26,0x0 which is "Invalid field in parameter list". If that is seen try again with the *−−non* option.

Downloading incorrect microcode into a device has the ability to render that device inoperable. One would hope that the device vendor verifies the data before activating it.

A long (operating system) timeout of 7200 seconds is set on each SEND DIAGNOSTIC command.

All numbers given with options are assumed to be decimal. Alternatively numerical values can be given in hexadecimal preceded by either "0x" or "0X" (or has a trailing "h" or "H").

EXAMPLES

If no microcode/firmware file is given then this utility fetches and decodes the Download microcode status dpage which could possibly show another initiator in the process of updating the microcode. Even if that is happening, fetching the status page should not cause any problems:

```
 sg_ses_microcode /dev/sg3
Download microcode status diagnostic page:
 number of secondary sub-enclosures: 0
 generation code: 0x0
  sub-enclosure identifier: 0 [primary]
    download microcode status: No download microcode operation in progress [0x0]
    download microcode additional status: 0x0
    download microcode maximum size: 1048576 bytes
    download microcode expected buffer id: 0x0
    download microcode expected buffer id offset: 0
```

The following sends new microcode/firmware to an enclosure. Sending a 1.5 MB file in one command caused the enclosure to lock up temporarily and did not update the firmware. Breaking the firmware file into 4 KB chunks (an educated guess) was more successful:

```
 sg_ses_microcode -b 4k -m dmc_offs_save -I firmware.bin /dev/sg4
```

The firmware update occurred in the following enclosure power cycle. With a modern enclosure the Extended Inquiry VPD page gives indications in which situations a firmware upgrade will take place.

EXIT STATUS

The exit status of sg_ses_microcode is 0 when it is successful. Otherwise see the sg3_utils(8) man page.

AUTHORS

Written by Douglas Gilbert.

REPORTING BUGS

Report bugs to <dgilbert at interlog dot com>.

COPYRIGHT

SEE ALSO

sg_ses, sg_write_buffer, sg_inq(sg3_utils)