SNMP Command Line Interface

for `scli` Version 0.3.1

by Jürgen Schönwälder

1 How to Read This Manual

This document contains two parts. The first part explains the design and the history of the `scli` tool. The second part provides a description of the modes and commands provided by `scli`. The third part contains a step by step instructions how `scli` can be extended with new modes and commands

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

1. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

2. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

3. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

4. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
   a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

5. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
6. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

7. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
8. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.
9. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
10. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
11. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software

which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

12. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
13. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

2 Overview

The GNU `scli` program implements a network management program that provides a command line interface to browse, monitor, and configure network devices. It uses the Simple Network Management Protocol (SNMP) to communicate with the devices and supports several standardized and some proprietary Management Information Base (MIB) modules.

Most SNMP management tools try to be very generic and they often fail to be useful. A good example are generic MIB browsers that display raw MIB data structures. These browsers tend to be of little use for actual management because MIB data structures are designed to be read and undertstood by programs rather than humans and thus difficult to deal with. Furthermore, the SNMP way of accessing MIB data is often simplistic and not optimized for humans. For example, humans usually prefer to refer to access objects via names rather than numbers while a machine oriented protocol such as SNMP generally prefers numbers over names.

The SNMP command line interface (`scli`) was designed to be specific rather than generic. In particular, `scli` understands the data it manipulates and it presents data in a way which is optimized for a human interface. It is not uncommon that `scli` uses information from different MIB modules in order to display data in a format which is easy to understand for human beings.

2.1 History

The `scli` package was written because of continued frustration how complex it is to configure and troubleshoot SNMP manageable devices with commonly used SNMP tools. The observation that better tools are needed to support SNMP management is nothing new. The author tried several approaches to tackle this problem before. In the 1990's, work was done on scripting language API extensions to simplify the interaction with networked devices. The result of these efforts was the Tnm [] extension for Tcl [], which is widely deployed these days, especially for testing and device emulation purposes.

In the 1990's, the author also experimented with concepts of network management platform, which resulted in `tkined`, a kind of a light-weight management platform. The `tkined` platform provides generic services and is highly extensible. In order to extend `tkined` with new management functionalities, all you need to do is to write a Tcl script. Although this sounds nice in theory, the reality is that most people who were interested in `tkined` did not want to write management scripts. Although the package has been openly available for many years, only few script contributions went back into the source distribution.

The author also wrote several generic MIB browsers. A very old one is still running on our Web server and being used regularly by people to browse MIBs and agents. But such Web-based solutions do not fit into into the preferred work style of the author. Firing up a complex Web browser just to retrieve and display a (potentially big) HTML page which shows management data at the abstraction level of an API between network elements in a distributed system just does not seem to be right and efficient.

In recent years, the `scli` author was involved in the specification and implementation of MIBs for the delegation of management functions. Very early in the project, it became clear that we need to have a good front-end. Again, we tried several approaches. The first one was a Java based applet running in a Web browser. This was usable for demos,

but nothing for everyday work. The second front-end was a Java stand-alone program. It provides a better user interface, but it eats up so many resources (and refuses to run on my 256 color display) so that people again used the old Web-based MIB browser which was written several years ago.

The conclusion one can derive from all these observations is that the approach to build generic tools is fundamentally flawed. Instead, it is necessary to build very specific tools which understand the management data they manipulate and which generate output which is optimized for humans. Furthermore, many people accept command line interface as a natural human interface for interacting with network elements.

2.2 Project Design

It is time consuming to build specific tools rather than generic tools. Someone who understands specific MIBs must design easy to use human management interfaces on top of them. The only reasonable way to tackle this problem with limited resources is to start an open source project and to get lots of programmers involved. This leads to the requirement that the source code must be modular so that specific extensions for e.g. modem management can be easily integrated.

The experience with past projects shows that the number of SNMP programmers is really not that huge. So this leads to another important requirement: It must be possible to invoke SNMP operations without any intimidate knowledge of the SNMP protocol and SNMP specific APIs. The technical mechanism to address this requirement is to use a MIB compiler which generates stub code for management applications from MIB modules. The stub code should export simple C structures that every programmer can easily understand and handle.

2.3 Software Architecture

3 Modes and Commands

The following sections describe the `scli` modes together with their commands.

100 scli version 0.4.0 (c) 2001-2010 Juergen Schoenwaelder 3COM MODE

The 3com scli mode allows to manipulate virtual lans (vlans) on 3com bridges. It is based on the PRODUCTMIB which is implemented at least on 3Com SuperStack II switches.

```
create 3com bridge vlan <vlanid> <name>
delete 3com bridge vlan <regexp>
set 3com bridge vlan name <vlanid> <name>
set 3com bridge vlan ports <regexp> <ports>
show 3com bridge vlan info [<regexp>]
dump 3com bridge vlan
```

The `create 3com bridge vlan` command is used to create a new virtual LAN with the given <vlanid> and <name>.

The `delete 3com bridge vlan` command deletes all selected virtual LANs. The regular expression <regexp> is matched against the virtual LAN names to select the vlans that should be deleted.

The `set 3com bridge vlan name` command changes the name of a virtual LAN.

The `set 3com bridge vlan ports` command allows to assign ports to port-based virtual LANs. The regular expression <regexp> is matched against the virtual LAN names to select the vlans that should be modified. The <ports> argument contains a comma separated list of port numbers or port number ranges, e.g. 1,5,7-8.

The `show 3com bridge vlan info` command shows summary information about all selected virtual LANs. The optional regular expression <regexp> is matched against the virtual LAN names to select the virtual LANs of interest. The command generates a table with the following columns:

```
VLAN     virtual LAN number
STATUS   status of the virutal LAN (see below)
NAME     name of the virutal LAN
INTERFCE virtual LAN interface number
PORTS    ports assigned to the virtual LAN
```

The status is encoded in two characters. The first character indicates the status of the row (A=active, S=not in service, R=not ready). The second character indicates virutal LAN type (P=port, I=IP-subnet, O=protocol, S=src address, D=dst address).

The `dump 3com bridge vlan` command generates a sequence of scli commands which can be used to restore the virtual LAN configuration.

3.1 ATM MODE

The atm scli mode is based on the ATM-MIB as published in RFC 2515. This mode is intended to display and configure ATM parameters.

```
show atm interface info <regexp>
show atm interface details <regexp>
```

The `show atm interface info` command displays summary information for all selected ATM interfaces. The optional regular expression <regexp> is matched against the interface descriptions to select the interfaces of interest. The command generates a table with the following columns:

```
INTERFACE   network interface number
DESCRIPTION description of the network interface
```

The `show atm interface details` command describes the selected ATM interfaces in more detail. The optional regular expression <regexp> is matched against the interface descriptions to select the interfaces of interest.

3.2 BRIDGE MODE

The scli bridge mode is based on the BRIDGE-MIB as published in RFC 4188 and the Q-BRIDGE-MIB as published in RFC 4363. It provides commands to browse information specific to IEEE 802.1 LAN bridges (also known as layer two switches).

```
show bridge info
show bridge ports
show bridge stp ports
show bridge forwarding
show bridge filter
show bridge stats
monitor bridge stats
show bridge vlan info [<regexp>]
show bridge vlan details [<regexp>]
create bridge vlan <id> <name>
delete bridge vlan <regexp>
```

The `show bridge info` command displays summary information about a bridge, such as the number of ports and the supported bridging functions and associated parameters.

The `show bridge ports` command displays information about the bridge ports.

The `show bridge stp ports` command displays information about the bridge ports which participate in the spanning tree protocol. The command generates a table with the following columns:

```
PORT     port number
PRIO     spanning tree priority of the port
STATE    spanning tree status of the port
P-COST   path costs for this port
D-ROOT   designated root port
D-COST   designated costs
D-BRIDGE designated bridge
D-PORT   designated port
```

The status is encoded in two characters. The first character indicates whether STP on the port is enabled (E) or disabled (D). The second character indicates the current status (D=disabled, B=blocking, I=listening, L=learning, F=forwarding, X=broken).

The `show bridge forwarding` command displays the forwarding data base used by transparent bridges. The command generates a table with the following columns:

```
PORT    port number
STATUS  status of the forwarding entry
ADDRESS address associated with the port
NAME    name of the address (where known)
VENDOR  vendor info derived from the address
```

The `show bridge filter` command shows filtering information.

The `show bridge stats` command displays per port statistics for transparent bridges. The command generates a table with the following columns:

```
PORT        port number
I-FPS       input frames per second
O-FPS       output frames per second
D-FPS       discarded frames per second
DESCRIPTION description of the port
```

The `monitor bridge stats` command shows the same information as the show bridge stats command. The information is updated periodically.

The `show bridge vlan info` command shows summary information about configured VLANs. The command generates a table with the following columns:

```
VLAN        VLAN number (between 1 and 4094)
STATUS         status of the VLAN
NAME        name of the VLAN
PORTS       ports assigned the the VLAN
```

The `show bridge vlan details` command describes the selected VLANs in detail. The optional regular expression <regexp> is matched against the VLAN names to select the VLANs of interest.

The `create bridge vlan` command creates a new vlan with the given <id> and name <name>.

The `delete bridge vlan` command deletes all vlans whose vlan name matches the regular expression <regexp>.

3.3 CISCO MODE

The cisco scli mode is used to display and configure cisco parameters. It also supports retrieval of accounting data from devices that support the old cisco accounting mib. This mode is based on the OLD-CISCO-IP-MIB published in May 1994.

```
show cisco processes
show cisco ip accounting info
show cisco ip accounting current sorted
show cisco ip accounting current raw
show cisco ip accounting snapshot sorted
show cisco ip accounting snapshot raw
monitor cisco ip accounting current
monitor cisco ip accounting snapshot sorted
set cisco ip accounting checkpoint
show cisco dot11 interface info
show cisco dot11 clients stats
```

```
monitor cisco dot11 clients stats
```

The `show cisco processes` command displays information about all processes running on a CISCO device. The command generates a table with the following columns:

```
CPU     processor executing a given process
PID     process indentification number on a CPU
P       priority of the process
MEMORY  memory used by the process
TIME    CPU time used by the process
COMMAND command executed by the process
```

The `show cisco ip accounting info` command displays general status information concerning the simple cisco IPv4 accounting mechanism supported by many older cisco devices. In particular, it displays the starting point of the current and snapshot data tables, information about the available accounting capacity, and statistics about lost bytes and packets.

cisco IP current accounting data

The `show cisco ip accounting current raw` command displays the raw accounting data retrieved from the current table. The command generates a table with the following columns:

```
SOURCE      source IPv4 address in dotted notation
DESTINATION destination IPv4 address in dotted notation
PACKETS     packets sent from source to destination
BYTES       bytes sent from source to destination
```

cisco IP snapshot accounting data

The `show cisco ip accounting snapshot raw` command displays the raw accounting data retrieved from the snapshot table. The command generates a table with the following columns:

```
SOURCE      source IPv4 address in dotted notation
DESTINATION destination IPv4 address in dotted notation
PACKETS     packets sent from source to destination
BYTES       bytes sent from source to destination
```

cisco IP current accounting data

cisco IP snapshot accounting data

The `set cisco ip accounting checkpoint` command takes a snapshot of the current accounting table by copying it to the snapshot accounting table. The current accounting table is reinitialized before it is updated again. The command returns the serial number of the snapshot.

The `show cisco dot11 interface info` command displays information about all IEEE 802.11 interfaces on a CISCO device. The command generates a table with the following columns:

```
IFACE  network interface number
SPEED  speed in bits per second
NAME   name of the network interface
CLNT   number of associated clients
BRDG   number of assiciated bridges
RPRT   number of assiciated repeaters
```

```
ASSCI  total number of associated stations
ASSCO  total number of deassociated stations
ROAMI  total number of roamed-in stations
ROAMO  total number of roamed-away stations
AUTHI  total number of authenticated stations
AUTHO  total number of deauthenticated stations
```

The `show cisco dot11 clients stats` command displays information about all IEEE 802.11 clients associated with a CISCO device. The command generates a table with the following columns:

```
IF               network interface number
SSID             SSID to which client is associated
ADDRESS          client MAC address
IPv4-ADDRESS  client}s IPv4 address (if supplied)
SGNL          client}s signal strength
UPTIME        lifetime of client}s association
I-BPS         input bytes per second
O-BPS         output bytes per second
ERR           errors per second
```

The `monitor cisco dot11 clients stats` command shows the same information as the show cisco dot11 clients stats command. The information is updated periodically.

3.4 DISMAN MODE

The scli disman mode is based on the DISMAN-SCRIPT-MIB as published in RFC 3165 and the DISMAN-SCHEDULE-MIB as published in RFC 3231. It allows to browse and configure distributed managers.

```
create disman script <owner> <name> <description>
create disman run <owner> <name> <args>
show disman languages
show disman script info
show disman script details
show disman launch info
show disman launch details
show disman run info
show disman run details
show schedule info
show schedule details
create schedule <owner> <name> <expression>
delete schedule <owner> <name>
dump schedule
monitor schedule info
monitor disman run
```

...

...

languages supported by the distributed manager

script summary information

scripts installed at the distributed manager

launch summary information

launch buttons installed on the distributed manager

summary information about running scripts

running scripts on the distributed manager

The `show schedule info` command displays summary information about the scheduled actions.

schedules on the distributed manager

...

...

The `dump schedule` command generates a sequence of scli commands which can be used to restore the schedule configuration.

scheduler information

monitor running scripts

3.5 ENTITY MODE

The entity scli mode is based on the ENTITY-MIB as published in RFC 2737. It provides commands to browse the physical entities or physical components that make up a managed system.

```
show entity info
show entity details
show entity containment
show entity sensors
```

The `show entity info` command displays summary information about the physical entities that compose the system. The command generates a table with the following columns:

```
ENTITY      entity number
CLASS       class of the entity (see below)
NAME        name of the entity
DESCRIPTION description of the entity
```

The `show entity details` command describes the physical entities in more detail.

The `show entity containment` command displays the physical entity containment hierarchy.

The `show entity sensors` command describes the physical sensor entities in more detail.

3.6 ETHERNET MODE

The ethernet scli mode is based on the EtherLike-MIB as published in RFC 2665 and the MAU-MIB as published in RFC 2668.

```
show ethernet mau
show ethernet stats
show ethernet history
```

```
monitor ethernet stats
```

The `show ethernet mau` command displays information about the medium attachment units (MAUs) for each ethernet port. The command generates a table which has the following columns:

```
INTERFACE network interface number
MAU       medium attachment unit number per interface
STATUS    status of the medium attachment unit
MEDIA     media availability
JABBER    jabber state of the medium attachment unit
AUTONEG   autonegation capabilities
TYPE      type of the medium attachment unit
```

The `show ethernet stats` command displays ethernet specific statistics for each ethernet interface. The command outputs a table which has the following columns:

```
INTERFACE network interface number
ALIGN     alignement errors per second
FCS       frame check sequence errors per second
RCV       MAC receive errors per second
LONG      frames exceeding maximum frame size per second
DEFER     deferred transmission per second
SCOL      single collisions per second
MCOLR     multiple collisions per second
XCOL      excessive collisions per second
LCOL      late collisions per second
XMIT      MAC transmit errors per second
CARR      carrier sense errors per second
...
```

The `monitor ethernet stats` command shows the same information as the show ethernet stats command. The information is updated periodically.

3.7 HP MODE

The hp scli mode is used to display and configure hp parameters.

```
show hp fault log
```

3.8 XXX

3.9 INTERFACE MODE

The scli interface mode is based on the IF-MIB as published in RFC 2863. It provides commands to browse, monitor and configure arbitrary network interfaces.

```
create interface stack <lower-regexp> <higher-regexp>
delete interface stack <lower-regexp> <higher-regexp>
set interface status <regexp> <status>
set interface alias <regexp> <string>
set interface notifications <regexp> <value>
```

```
set interface promiscuous <regexp> <bool>
show interface info [<regexp>]
show interface details [<regexp>]
show interface stack [<regexp>]
show interface stats [<regexp>]
monitor interface stats [<regexp>]
loop interface stats [<regexp>]
check interface status [<regexp>]
dump interface
```

The `set interface status` command modifies the administrative status of all selected interfaces. The regular expression <regexp> is matched against the interface descriptions to select the interfaces of interest. The <value> parameter must be one of the strings "up", "down", or "testing".

The `set interface alias` command assigns the alias name <string> to the selected interfaces. The alias name provies a non-volatile handle which can be used by management applications to better identify interfaces. The regular expression <regexp> is matched against the interface descriptions to select the interfaces.

The `set interface notifications` command controls whether the selected interfaces generate linkUp and linkDown notifications. The regular expression <regexp> is matched against the interface descriptions to select the interfaces. The <value> parameter must be one of the strings "enabled" or "disabled".

The `set interface promiscuous` command controls whether the selected interfaces operate in promiscuous mode or not. The regular expression <regexp> is matched against the interface descriptions to select the interfaces. The <bool> parameter must be one of the strings "true" or "false".

The `show interface info` command displays summary information for all selected interfaces. The optional regular expression <regexp> is matched against the interface descriptions to select the interfaces of interest. The command generates a table with the following columns:

```
INTERFACE   network interface number
STATUS      interface status (see below)
MTU         maximum transfer unit
TYPE        type of the network interface
SPEED       speed in bits per second
NAME        name of the network interface
DESCRIPTION description of the network interface
```

The status is encoded in four characters. The first character indicates the administrative status (U=up, D=down, T=testing). The second character indicates the operational status (U=up, D=down, T=testing, ?=unknown, O=dormant, N=not-present, L=lower-layer-down). The third character indicates whether a connector is present (C=connector, N=none) and the fourth character indicates whether the interface is in promiscuous mode (P=promiscuous, N=normal).

The `show interface details` command describes the selected interfaces in detail. The optional regular expression <regexp> is matched against the interface descriptions to select the interfaces of interest.

The `show interface stack` command shows the stacking order of the interfaces. The command generates a table with the following columns:

```
INTERFACE   network interface number
STACK       indication of the stacking order
TYPE        type of the network interface
DESCRIPTION description of the network interface
```

The `show interface stats` command displays network interface statistics for all selected interfaces. The optional regular expression <regexp> is matched against the interface description to select the interfaces. The command outputs a table which has the following columns:

```
INTERFACE   network interface number
STATUS      interface status (see above)
I-BPS       input bytes per second
O-BPS       output bytes per second
I-PPS       input packets per second
O-PPS       output packets per second
I-ERR       input errors per second
O-ERR       output errors per second
I-DIS       input packets discarded per second
O-DIS       output packets discarded per second
I-UNK       input packets with unknown protocols per second
DESCRIPTION description of the network interface
```

The `monitor interface stats` command shows the same information as the show interface stats command. The information is updated periodically.

The `loop interface stats` command shows the same information as the show interface stats command. The information is updated periodically.

The `check interface status` command checks the status of interfaces. The optional regular expression <regexp> is matched against the interface description to select the interfaces In particular, the check interface status commands detects fault conditions if (a) ifAdminStatus is not down and ifOperStatus is down or (b) ifAdminStatus is down and ifOperStatus is not down and not notPresent

The `dump interface` command generates a sequence of scli commands which can be used to restore the interface configuration.

3.10 IP MODE

The ip scli mode is based on the IP-MIB as published in RFC 2011, the IP-FORWARD-MIB as published in RFC 2096, the IP-TUNNEL-MIB as published in RFC 2667 and the RFC1213-MIB as published in RFC 1213. It provides commands to browse, monitor and configure IP protocol engines.

```
set ip forwarding <value>
set ip ttl <number>
show ip info
show ip forwarding
show ip addresses
```

```
show ip tunnel
show ip mapping
dump ip
```

The `set ip forwarding` command controls whether the IP protocol engine forwards IP datagrams or not. The <value> parameter must be one of the strings "enabled" or "disabled".

The `set ip ttl` command can be used to change the default time to live (TTL) value used by the IP protocol engine. The <number> parameter must be a number between 1 and 255 inclusive.

The `show ip info` command displays parameters of the IP protocol engine, such as the default TTL or whether the node is forwarding IP packets.

The `show ip forwarding` command displays the IP forwarding data base. The command generates a table with the following columns:

```
DESTINATION destination address and prefix
NEXT-HOP    next hop towards the destination
TOS         type of service selector
TYPE        type (direct/indirect) of the entry
PROTO       protocol which created the entry
INTERFACE   interface used for forwarding
```

The `show ip addresses` command displays the IP addresses assigned to network interfaces. The command generates a table with the following columns:

```
ADDRESS     IP address
PREFIX      IP address prefix length
NAME        name of the IP address
INTERFACE   network interface number
DESCRIPTION description of the network interface
```

The `show ip tunnel` command displays information about existing IP tunnels.

The `show ip mapping` command displays the mapping of IP address to lower layer address (e.g., IEEE 802 addresses). The command generates a table with the following columns:

```
INTERFACE network interface number
STATUS    status of the mapping entry
ADDRESS   IP address
ADDRESS   lower layer address
```

The `dump ip` command generates a sequence of scli commands which can be used to restore the IP configuration.

3.11 ISDN MODE

The scli isdn mode is based on the ISDN-MIB as published in RFC 2127.

```
show isdn bri [<regexp>]
show isdn bearer
show isdn endpoints
```

The `show isdn bri` command shows information about the ISDN basic rate interfaces. The command outputs a table which has the following columns:

```
INTERFACE   network interface number
TYPE        type of the ISDN interface
TOPOLOGY    line topology
MODE        interface mode (te/nt)
SIGNALING   signaling mode (active/inactive)
DESCRIPTION description of the network interface
```

The `show isdn bearer` command shows information about the ISDN B (bearer) channels.

The `show isdn endpoints` command shows information about the ISDN endpoints.

3.12 NETSNMP MODE

The netsnmp scli mode is used to display and configure netsnmp specific parameters. It is based on the UCD-SNMP-MIB.

```
set netsnmp debugging <value>
set netsnmp restart
show netsnmp info
show netsnmp load
show netsnmp exec
show netsnmp proc
dump netsnmp
```

The `set netsnmp debugging` command controls whether the agent generates debug messages or not. The <value> parameter must be one of the strings "enabled" or "disabled".

The `set netsnmp restart` command restarts the agent.

The `show netsnmp info` command shows general information about the netsnmp/ucdsnmp agent such as the version number and the software configuration.

The `show netsnmp load` command shows the load indices of the system. This is usually the length of the queue in front of the processor(s) averaged over some time interval.

The `show netsnmp exec` command shows information about pre-configured commands that can be invoked.

The `show netsnmp proc` command shows information about which processes netsnmp watches.

The `dump netsnmp` command generates a sequence of scli commands which can be used to restore the netsnmp specific configuration.

3.13 NORTEL MODE

The nortel scli mode allows to manipulate virtual LANs (vlans) on nortel bridges. It is based on the RAPID-CITY MIB which is implemented at least on the baystack bridges.

```
create nortel bridge vlan <vlanid> <name>
delete nortel bridge vlan <regexp>
set nortel bridge vlan name <vlanid> <name>
set nortel bridge vlan ports <regexp> <ports>
set nortel bridge vlan default <string> <ports>
show nortel bridge vlan info [<regexp>]
```

```
show nortel bridge vlan details [<regexp>]
show nortel bridge vlan ports
dump nortel bridge vlan
```

The `create nortel bridge vlan` command is used to create a new virtual LAN with the given <vlanid> and <name>.

The `delete nortel bridge vlan` command deletes all selected virtual LANs. The regular expression <regexp> is matched against the virtual LAN names to select the vlans that should be deleted.

The `set nortel bridge vlan name` command changes the name of a virtual LAN.

The `set nortel bridge vlan ports` command allows to assign ports to port-based vlans. The regular expression <regexp> is matched against the vlan names to select the vlans that should be modified. The <ports> argument contains a comma separated list of port numbers or port number ranges, e.g. 1,5,7-8.

The `set nortel bridge vlan default` command allows to assign ports to a default vlan. The <string> argument is matched against the vlan names to select the vlan. The <ports> argument contains a comma separated list of port numbers or port number ranges, e.g. 1,5,7-8.

The `show nortel bridge vlan info` command shows summary information about all selected virtual LANs. The optional regular expression <regexp> is matched against the virtual LAN names to select the virtual LANs of interest. The command generates a table with the following columns:

```
VLAN   number of the virtual LAN
STATUS status of the virtual LAN (see below)
NAME   name of the virtual LAN
PORTS  ports assigned to the virtual LAN
```

The status is encoded in four characters. The first character indicates the status of the row (A=active, S=not in service, R=not ready). The second character indicates virtual LAN type (P=port, I=IP-subnet, O=protocol, S=src address, D=dst address). The third character indicates the priority of the virtual LAN (H=high, N=normal) and the fourth character indicates whether routing is enabled (R=routing, N=no routing).

The `show nortel bridge vlan details` command describes the selected vlans in more detail. The optional regular expression <regexp> is matched against the vlan names to select the vlans of interest.

The `show nortel bridge vlan ports` command shows information for each vlan port. The command generates a table with the following columns:

```
PORT    port number
FLAGS   port vlan flags (see below)
DEFAULT default vlan number
VLANS   vlan numbers the port is member of
```

The flags are encoded in four characters. The first character indicates the port type (A=access, T=trunk). The second character indicates whether the port tags frames (T=tagging, N=none). The third character indicates whether the port discards tagged frames (D=discard, N=none) and the fourth character indicates whether the port discards untagged frames (D=discard, N=none).

The `dump nortel bridge vlan` command generates a sequence of scli commands which can be used to restore the virtual LAN configuration.

3.14 OSPF MODE

The scli ospf mode is used to display and configure OSPF parameters.

```
show ospf area
show ospf info
show ospf interface
show ospf lsdb
```

show OSPF areas

general OSPF information

show OSPF interfaces

show OSPF lsdb

3.15 PRINTER MODE

The scli printer mode is based on the Printer-MIB as published in RFC 1759 and some updates currently being worked on in the IETF Printer MIB working group.

```
set printer operator <string>
show printer info
show printer paths
show printer inputs
show printer outputs
show printer markers
show printer colorants
show printer supplies
show printer interpreters
show printer channels
show printer covers
show printer display
show printer lights
show printer alerts
monitor printer display
monitor printer lights
monitor printer alerts
run printer reboot
```

The `set printer operator` command configures the name of the person responsible for operating a printer. As a convention, the phone number, fax number or email address should be indicated by the tel:, fax: and mailto: URL schemes.

The `show printer info` command shows general information about the printer including global status information.

The `show printer paths` command shows information about the media paths of a printer.

The `show printer inputs` command shows information about the input sub-units of a printer which provide media for input to the printing process.

The `show printer output` command shows information about the output sub-units of a printer capable of receiving media delivered from the printing process.

The `show printer markers` command shows information about the marker sub-units of a printer which produce marks on the print media.

The `show printer colorants` command shows information about the colorant sub-units of a printer which produce marks on the print media.

The `show printer supplies` command shows information about the supplies which are consumed and the waste produced by the markers of a printer.

The `show printer interpreters` command shows information about the page description language and control language interpreters supported by the printer.

The `show printer channels` command shows information about the channels which can be used to submit data to the printer.

The `show printer covers` command shows information about the covers of a printer.

The `show printer display` command shows the current contents of the display attached to the printer. The command generates a table with the following columns:

```
PRINTER logical printer number
LINE    display line number
TEXT    contents of the display line
```

The `show printer lights` command shows the current status of the lights attached to the printer. The command generates a table with the following columns:

```
PRINTER     logical printer number
LIGHT       number identifying the light/led
DESCRIPTION description of the light/led
STATUS      current status (on, off, blink)
COLOR       current color of the light
```

The `show printer alerts` command displays the list of active printer alerts including the alert code, the alert severity, the alert description, the alert time, the alert location and the personel required to handle the alert.

The `monitor printer display` command shows the same information as the show printer display command. The information is updated periodically.

The `monitor printer lights` command shows the same information as the show printer lights command. The information is updated periodically.

The `monitor printer alerts` command shows the same information as the show printer alerts command. The information is updated periodically.

The `run printer reboot` command resets the printed. RS232 MODE

The rs232 scli mode is based on the RS-232-MIB as published in RFC 1659.

```
show rs232 details
```

The `show rs232 details` command describes the selected RS 232 interfaces in detail.

3.16 SCLI MODE

The scli mode provides commands that can be used to display and manipulate the internal state of the scli interpreter.

```
open <nodename> [<community>]
close
run scli walk <oid> [<oid> ...]
run scli scan <network> [community]
run scli sleep <secs>
create scli plugin <module>
delete scli plugin <module>
exit
help
history
create scli alias <name> <value>
delete scli alias <regexp>
create scli interp <name>
delete scli interp <regexp>
set scli regex [<regexp>]
set scli debugging [<regexp>]
set scli pager <pager>
set scli retries <retries>
set scli timeout <milliseconds>
set scli format <fmt>
set scli mode <mode>
show scli info
show scli command info [<regex>]
show scli command details [<regex>]
show scli command tree
show scli aliases
show scli modes [<regex>]
show scli schema [<regex>]
show scli alarm info
```

The `open` command establishes an association to a remote SNMP agent. The <nodename> argument is the DNS name or the IP address of the remote node. Scli will try to talk to the SNMP agent on this node by using the default port number (usually 161) and the default transport mapping (usually SNMP over UDP). The optional <community> argument is the community string needed to communicate with the remote SNMP agent. The default community string is "public". Opening an association while an association is already established is not considered an error. The existing established association will be closed automatically before an attempt to create a new association is started.

The `close` command closes an established association to a remote SNMP agent. Invoking the close command when no association is established is not considered an error and will do just nothing.

The `run scli walk` command is a debugging utility which simply performs a MIB walk. Note that scli does not have general MIB knowledge and hence the output requires some post-processing.

The `run scli scan` command is a utility which scans an IPv4 address space identified by the <network> argument. The <network> must be specified in the format <ipv4address>/<prefix>. The optional <community> argument is the community string needed to communicate with the remote SNMP agent. The default community string is "public".

The `run scli sleep` command simply sleeps for the given amount of seconds.

The `create scli plugin` command dynamically loads an scli mode into a running scli process. This can be used to dynamically extend scli with modules coming from other sources. Dynamic loadable modules also simplify the development and management of site-specific modules.

The `delete scli plugin` command removes a previously loaded modules from a running scli process.

The `exit` command terminates the scli interpreter. An end of file in the standard input stream will also terminate the the scli interpreter.

The `help` command displays some help information including a list of all top-level scli commands.

The `history` command displays the scli command history list with line numbers.

The `create scli alias` command creates the alias <name> for the scli command (fragment) <value>. If the alias <name> already exists, then the new <value> will be assigned to the existing alias.

The `delete scli alias` command removes previously defined aliases from the scli interpreter. The regular expression <regexp> is matched against all alias names in order to select the aliases that are deleted.

The `create scli interp` command creates a new internal scli interpreter with the name <name>.

The `delete scli interp` command deletes previously defined scli interpreters from the main scli interpreter. The regular expression <regexp> is matched against all alias names in order to select the interpreter(s) to be removed.

The `set scli regex` command controls how scli matches regular expressions. The optional regular expression <regexp> is matched against the regular expression options. A successful match turns a regular expression option on while an unsuccessful match turns a regular expression option off. Invoking the command without the <regexp> argument will turn all regular expression options off. The currently defined regular expression options are "extended" for POSIX extended regular expressions and "case-insensitive" for case insensitive matches.

The `set scli debugging` command sets the debugging level of the SNMP engine. The optional regular expression <regexp> is matched against the debugging levels. A successful match turns a debugging level on while an unsuccessful match turns a debugging level off. Invoking the command without the <regexp> argument will turn all debugging levels off. The currently defined debugging levels are "session" for the SNMP session layer, "request"

for the SNMP request handling layer, "transport" for the SNMP transport layer, "packet" for the SNMP packet layer, and "asn1" for the ASN.1 coding layer.

The `set scli pager` command defines the shell command which is used as a pager if the output produced by an scli command does not fit on a single screen. The output is passed to the <pager> shell command via its standard input stream.

The `set scli retries` command defines the number of SNMP request retries before giving up requesting a certain object.

The `set scli timeout` command defines the number milliseconds between subsequent SNMP request retries.

The `set scli format` command defines the output format used by subsequent scli commands. The currently supported formats are "scli" and "xml". The "scli" format is the default output format and described in this documentation. The "xml" output format is experimental and therefore not described here.

The `set scli mode` command defines the scli mode used by subsequent scli commands. Setting the mode to "protocol" will force scli to work in protocol mode. Setting the mode to "normal" causes scli to work in normal mode where certain status messages are suppressed.

The `show scli info` command displays the current status of the scli interpreter.

The `show scli command info` command displays summary information about scli commands. The optional regular expression <regex> is matched against the command names to select the scli commands.

The `show scli command details` command displays detailed information about scli commands. The optional regular expression <regex> is matched against the command names to select the scli commands.

The `show scli command tree` command displays the scli command tree. The full command syntax is displayed for each leaf node.

The `show scli aliases` command lists all scli command aliases. The first column in the generated table lists the aliase names while the second column shows the alias values.

The `show scli modes` command shows information about the scli modes. An scli mode is a logical grouping of related commands (e.g., all commands that deal with printers). The optional regular expression <regex> can be use to select a subset of the available scli modes.

The `show scli schema` command produces xml schema definitions for the selected scli modes. An scli mode is a logical grouping of related commands (e.g., all commands that deal with printers). The optional regular expression <regex> can be use to select a subset of the available scli modes.

The `show scli alarm info` command displays summary information about all known alarms.

3.17 SNMP MODE

The snmp scli mode is based on the SNMPv2-MIB as published in RFC 1907, the SNMP-FRAMEWORK-MIB as published in RFC 3411, the SNMP-USER-BASED-SM-MIB as published in RFC 3414, the SNMP-VIEW-BASED-ACM-MIB as published in RFC 3415, the SNMP-TARGET-MIB as published in RFC 3413, the SNMP-NOTIFICATION-MIB as published in RFC 3413, and theNOTIFICATION-LOG-MIB as published in RFC 3014.

```
create snmp vacm member <name> <group> [<model>]
delete snmp vacm member <regex-name> <regex-group> [<model>]
create snmp usm user <name> <template>
set snmp authentication traps <status>
show snmp engine
show snmp resources
show snmp vacm member
show snmp vacm access
show snmp vacm views
show snmp usm users
show snmp target addresses
show snmp target parameters
show snmp notification targets
show snmp contexts
show snmp csm
show snmp notification log details
show snmp notification log info
monitor snmp notification log info
dump snmp
```

The `create snmp vacm member` commands can be used to assign new members (security names) to vacm groups. New groups are created if they do not exist.

The `delete snmp vacm member` commands can be used to delete members (security names) from vacm groups. Groups are deleted if the last member is deleted.

The `create snmp usm user` commands can be used to create a new user by cloning an existing template.

The `set snmp authentication traps` command controls whether the SNMP engine generates authentication failure notifications. The <value> parameter must be one of the strings "enabled" or "disabled".

The `show snmp engine` command displays information about the SNMP protocol engine such as the number of boots, the current time relative to the last boot and the maximum message size.

The `show snmp resources` command displays information about the MIB resources supported by the SNMP agent.

The `show snmp vacm member` command displays the mapping of security names to group names. The command generates a table with the following columns:

```
ROW    row storage type and status
MOD    security model
NAME   member name (security name)
GROUP  name of the vacm group
```

The `show snmp vacm access` command display the access control rules for the security groups. The command generates a table with the following columns:

```
ROW    row storage type and status
GROUP  security group name
MOD    security model
LVL    security level (--, a-, ap)
```

```
CTX    context name
MATCH  match (exact or prefix)
READ   view name for read access
WRITE  view name for write access
NOTIFY view name for notification
```

The `show snmp vacm views` command displays MIB view definitions. The command generates a table with the following columns:

```
ROW    row storage type and status
VIEW   view name
TYPE   access to the view subtree (incl/excl)
PREFIX object identifier wildcard prefix
```

The `show snmp usm users` command displays the configured users. The command generates a table with the following columns:

```
ROW    row storage type and status
USER   USM user name
NAME   security name of the USM user
AUTH   authentication protocol
PRIV   privacy protocol
```

The `show snmp target addresses` command displays information about the configured SNMP target addresses. The command generates a table with the following columns:

```
ROW      row storage type and status
TARGET   target name
DOMAIN   transport domain
ADDRESS  transport address
TMOUT    timeout value in ms
RETRY    number of retries
PARAMS   associated parameters
TAGS     tag list
```

The `show snmp target parameters` command displays information about the configured SNMP target parameters. The command generates a table with the following columns:

```
ROW      row storage type and status
PARAMS   parameter name
NAME     security name
```

The `show snmp notification targets` command displays information about the configured SNMP notification targets. The command generates a table with the following columns:

```
ROW      row storage type and status
NAME     notification target name
TYPE     notification type
TAG      tag reference to targets
```

The `show snmp contexts` command displays information about the available SNMP contexts.

The `show snmp csm communities` command displays information about the configured SNMP communities.

The `show snmp notification log details` command displays detailed information about logged SNMP notifications.

The `show snmp notification log info` command displays summary information about logged SNMP notifications.

The `monitor snmp notification log info` command displays summary information about logged SNMP notifications.

The `dump snmp` command generates a sequence of scli commands which can be used to restore the engine configuration.

3.18 SONET MODE

The sonet scli mode is based on the SONET-MIB as published in RFC 2558. It provides commands to manage Synchronous Optical Network/Synchronous Digital Hierarchy (SONET/SDH) interfaces.

```
show sonet media [<regexp>]
show sonet section stats [<regexp>]
show sonet section history [<regexp>]
monitor sonet section stats [<regexp>]
```

The `show sonet media` command displays information about the configuration of SONET/SDH interfaces. The command outputs a table which has the following columns:

```
INTERFACE   network interface number
SIGNAL      type of the signal (SONET/SDH)
CODING      line coding (B3ZS, CMI, NRZ, RZ)
LINE        optical or electrical line type
DESCRIPTION description of the network interface
```

The `show sonet section stats` command displays statistics about SONET/SDH section errors. The command outputs a table which has the following columns:

```
INTERFACE   network interface number
INTERVAL    measurement interval
ES          errored seconds
SES         severely errored seconds
SEFS        severely errored framing seconds
CV          coding violations
LOSS        flags indicating loss of signal/frame
DESCRIPTION description of the network interface
```

The `show sonet section history` command displays 15 minute history statistics about SONET/SDH section errors. The command outputs a table which has the following columns:

```
INTERFACE   network interface number
INTERVAL    measurement interval start offset
ES          errored seconds
SES         severely errored seconds
SEFS        severely errored framing seconds
CV          coding violations
DESCRIPTION description of the network interface
```

The `monitor sonet section stats` command shows the same information as the show sonet section stats command. The information is updated periodically.

3.19 SYSTEM MODE

The system scli mode is primarily based on the SNMPv2-MIB as published in RFC 1907 and the HOST-RESOURCES-MIB as publisched in RFC 2790. It can be used to browse and configure system parameters and characteristics.

```
set system contact <string>
set system name <string>
set system location <string>
show system info
show system devices
show system storage
show system mounts
show system processes [<regexp>]
show system software [<regexp>]
monitor system storage
monitor system processes [<regexp>]
check system contact
check system storage
check system process [<regexp>] [<n>*<m>]
dump system
```

The `set system contact` command configures the system contact information. The <string> argument should include information on how to contact a person who is responsible for this system.

The `set system name` command configures the name of the system. By convention, this is the fully-qualified domain name.

The `set system location` command configures the physical location of the system.

The `show system info` command shows general information about the system.

The `show system devices` command shows a list of system devices. The command generates a table with the following columns:

```
INDEX       device number
STATUS      current status of the device
DESCRIPTION description of the device
```

The `show system storage` command displays information about the logical areas attached in the system. The command generates a table with the following columns:

```
INDEX       logical storage area number
DESCRIPTION description of the storage area
TYPE        logical storage area type
SIZE        total size of the storage area
USED        amount of storage in use
FREE        amount of storage available
USE%        used storage in percent
```

The `show system mounts` command shows the list of filesystems mounted on the system. The command generates a table with the following columns:

```
INDEX   filesystem identification number
LOCAL   local root path name of the filesystem
REMOTE  remote server and root path name (if any)
TYPE    filesytem type (if known)
OPTIONS access mode (ro/rw) and boot flag
```

The `show system processes` command display information about the processes currently running on the system. The regular expression <regexp> is matched against the command executed by the process to select the processes of interest. The command generates a table with the following columns:

```
PID     process identification number
S       status of the process (see below)
T       type of the process (see below)
MEMORY  memory used by the process
TIME    CPU time used by the process
COMMAND command executed by the process
```

The process status values are C=running, R=runnable, S=not runnable, and Z=invalid. The process types values are ?=unknown, O=operating system, D=device driver, and A=application.

The `show system software` command display information about the software installed on the system. The regular expression <regexp> is matched against the software name to select the software of interest. The command generates a table with the following columns:

```
SID     software identification number
T       type of the software (see below)
DATE    software installation date
NAME    software name
```

The software type values are ?=unknown, O=operating system, D=device driver, and A=application.

The `monitor system storage` command shows the same information as the show system storage command. The information is updated periodically.

The `monitor system processes` command show the same information as the show system processes command. The information is updated periodically.

The `check system contact` command xxx.

The `check system storage` command checks xxx.

The `check system process` command checks xxx.

The `dump system` command generates a sequence of scli commands which can be used to restore the system configuration.

3.20 TCP MODE

The scli tcp mode is based on the TCP-MIB as published in RFC 2012. It provides commands to browse information specific to the TCP transport protocol.

```
show tcp info
show tcp listener
show tcp connections
show tcp states
monitor tcp connections
monitor tcp states
```

The `show tcp info` command displays parameters of the TCP protocol engine.

The `show tcp listener` command displays the listening TCP endpoints. The command generates a table with the following columns:

```
LOCAL  local TCP endpoint
STATE  transmission control block state (listen)
```

The `show tcp connections` command displays the connected TCP endpoints including the current state of the connection as seen by the remote SNMP agent. The command generates a table with the following columns:

```
LOCAL  local  TCP endpoint
REMOTE remote TCP endpoint
STATE  transmission control block state
```

The transmission control block state is either closed, synSent, synReceived, established, finWait1, finWait2, closeWait, lastAck, closing, or timeWait.

The `show tcp states` command displays the distribution of TCP transmission control block states together with a list of known port names in each state. The command generates a table with the following columns:

```
COUNT number of transmission control blocks per state
STATE transmission control block state
PORTS well-known ports associated with the state
```

The command uses some heuristics to identify the interesting port numbers. First, all local port numbers are considered where the local port number matches one of the listening port numbers. From the remaining connections, all local port numbers are considered with a well known name. From the remaining connections, all remote port numbers are considered with a well known name. All remaining connections are aggregated under the pseudo name - (hyphen). Unspecified port numbers are show using the pseudo name * (star).

The `monitor tcp connections` command displays the connected TCP endpoints including the current state of the connection as seen by the remote SNMP agent. The information is updated periodically.

The `monitor tcp states` command displays the distribution of TCP connection states. The information is updated periodically.

3.21 UDP MODE

The scli sflow mode is based on the SFLOW-MIB as published in RFC 3176 and the SFLOW5-MIB published on the slow.org web site. It provides commands to browse information specific to sflow probes.

```
show sflow info
show sflow receiver
```

The `show sflow info` command displays information about an sflow implementation.

The `show sflow receiver` command displays information about sflow receivers.

3.22 UDP MODE

The scli udp mode is based on the UDP-MIB as published in RFC 2013. It provides commands to browse information specific to the UDP transport protocol.

```
show udp listener
show udp stats
```

The `show udp listener` command displays the listening UDP endpoints.

The `show udp statistics` about datagrams received or sent.

4 Writing Extensions

This chapter describes the process of extending scli with new commands. This is relatively straight-forward and does not require any SNMP programming skills. All you need is to be able to read and understand MIB definitions and some C programming skills.

The first thing you need to do in order to extend scli is to select the set of MIBs needed to implement the new commands. Once you know the list of required MIBs, you need to check whether there are already stubs for these MIBs in the `stub` directory. If some are missing, you have to create them.

Once you have all the MIB stubs, you need to decide which scli mode should contain your new commands. An scli mode is simply a logical grouping of related commands (e.g. all commands that deal with layer 2 bridges). A mode is usually implemented by a single C file in the `scli` directory which is named like the mode itself.

Now that you know the mode for you commands, you can start writing commands. Every command is implemented by a C function which basically has three parts: command syntax parsing, executing the command (usually this requires calls to other internal functions), and finally cleanup.

4.1 Creating Stubs

Compiler generated MIB stubs are used to do all the low-level SNMP work for you. To generate MIB stubs, you need to install the `smidump` MIB compiler which comes with the `libsmi` distribution. You need to make sure that the `smidump` compiler version fits with the `scli` version since there are sometimes changes between the stub interface and the SNMP engine interface. The easiest way to ensure that you have a suitable compiler is to check that the compiler version number matches the scli version number contained in comments in the stub files contained in the `scli` distribution.

4.2 Implementing Modes

Every scli mode implementation has only one externally visible entry point which is used to initialize the mode. This name of the entry point is by convention derived from the mode name. If we want to implement a mode "example", the function would look like this:

```
void scli_init_example_mode(scli_interp_t *interp)
```

The declaration of the function prototype goes into the file '`scli.h`' and a call of the initialization function must be added in the file '`scli.c`'.

The main purpose of the entry point is to register the mode. This is accomplished by filling out an instance of type `scli_mode_t` which describes the mode. The first member of the structure is the unique name of the mode. The second member is a string describing the purpose of the mode and which standards or proprietary MIB modules are relevant for it. The third parameter is a NULL-terminated vector of `scli_cmd_t` structures, one structure for each command.

```
void
scli_init_example_mode(scli_interp_t *interp)
{
    static scli_mode_t example_mode = {
```

```
        "example",
        "The scli example mode is only used to demontrate how you can\n"
        "extend the scli interpreter with new modes and commands.",
        NULL
    };

    scli_register_mode(interp, &printer_mode);
}
```

4.3 Implementing Commands

The function below implements a `show hello world` command. A function implementing an `scli` command is usually referred to as a *command procedure*. Every command procedure is called with three arguments:

```
int (*func) (scli_interp_t *interp, int argc, char **argv);
```

1. A pointer to the scli interpreter in *interp*.
2. The number of arguments in *argc*.
3. The command arguments as a vector of C strings in *argv*.

Command procedures return an `scli` status code. See the file '`scli.h`' for a description of the various error codes supported by the `scli` interpreter. Below is an example which implements a `show hello world` command.

```
static int
show_hello_world(scli_interp_t *interp, int argc, char **argv)
{
    if (argc > 1) {
        return SCLI_SYNTAX_NUMARGS;
    }

    if (scli_interp_dry(interp)) {
        return SCLI_OK;
    }

    g_string_sprintfa(interp->result, "hello world");

    return SCLI_OK;
}
```

The first thing we have to do in a command procedure is syntax parsing. This is important to get this right to ensure stability. In the example above, we simply check that no arguments are present. If there are any arguments, the command procedure returns the `scli` error code `SCLI_SYNTAX_NUMARGS`.

Once we have successfully parsed the command, we have to decide whether we really have to execute it. If the interpreter is running in dry mode, we just parse the commands but do not execute them. This is a handy tool for validating the syntax of an `scli` script file without actually touching any devices and can be used to test that your syntax parsing code really works as it should.

The next part of every command procedure implements the functionality. In this case, we just copy the string `"hello world"` into the interpreter's result string. The result string is a `glib` `GString` which can dynamically grow. You should study the `glib` documentation on how to use `GString`s effectively.

The last part in our command procedure is responsible for freeing any resources and to return the status code. Since the `show hello world` command always succeeds, we simple return `SCLI_OK`.

Once we have written a command procedure, we need to register the command in the interpreter. This is done by filling out a data structure which describes our command and registering this data structure as part of the mode registration.

4.4 Adding MIB Stubs

It might be necessary to add stubs for additional MIB modules. This process is very straightforward.

1. Edit `stub/Makefile.am`
2. Run `make` in the `stub` directory. Note that you need a version of `smidump` which is compatible with the `scli` release you are using.
3. Edit '`scli/scli.h`' by adding the new header file(s) to the includes at the end of the file.

4.5 Dynamic Loading (Plugins)

This chapter explains how you can add functionality to `scli` programs.

5 Problems

If you find a bug in `scli`, please send electronic mail to '`scli@ibr.cs.tu-bs.de`'. Include the version number, which you can find by running '`scli --version`'. Also include in your message the output that the program produced and the output you expected.

If you have other questions, comments or suggestions about `scli`, contact the author via electronic mail to '`schoenw@ibr.cs.tu-bs.de`'. The author will try to help you out, although he may not have time to fix your problems.

Concept Index

Short Contents

Table of Contents