

Ethernet switches

## **MES23xx, MES33xx, MES35xx, MES5324**

MES Ethernet switches monitoring and configuration via SNMP,  
firmware version 4.0.22

Document Version	Issue Date	Revisions
Version 1.25	07.03.2024	Synchronization with firmware version 4.0.22
Version 1.24	19.12.2023	Synchronization with firmware version 4.0.21.7
Version 1.23	20.10.2023	Synchronization with firmware version 4.0.21.5
Version 1.22	09.08.2023	Synchronization with firmware version 4.0.21  Sections changed: 3.6 Firmware update 4.1 System resources 10.2 Spanning-tree protocol 12.1 AAA mechanism 16.5 Port-based client authentication (802.1x)  Sections added: 3.3 Operation with SFTP and SCP servers 3.4 Downloading a license file to a device 6.6 Configuring Flex-link
Version 1.21	28.04.2023	Synchronization with firmware version 4.0.20
Version 1.20	20.01.2023	Synchronization with firmware version 4.0.19  Sections changed: 3.1 Saving the configuration 3.2 Operation with TFTP server 3.6 Firmware update
Version 1.19	16.12.2022	Synchronization with firmware version 4.0.18.4
Version 1.18	29.11.2022	Synchronization with firmware version 4.0.18.2
Version 1.17	29.07.2022	Synchronization with firmware version 4.0.18  Sections changed: 10.2 Spanning-tree protocol
Version 1.16	27.06.2022	Synchronization with firmware version 4.0.17  Sections changed: 4.1 System resources 4.2 System parameters 6.2 Configuring VLAN 13 Port Mirroring 14.2 Optical transceiver diagnostics
Version 1.15	01.04.2022	Synchronization with firmware version 4.0.16.14
Version 1.14	28.02.2022	Synchronization with firmware version 4.0.16.13
Version 1.13	12.01.2022	Synchronization with firmware version 4.0.16.11
Version 1.12	12.11.2021	Synchronization with firmware version 4.0.16.5
Version 1.11	12.10.2021	Synchronization with firmware version 4.0.16.4
Version 1.10	31.07.2021	Synchronization with firmware version 4.0.16.2
Version 1.9	30.03.2021	Sections changed: 19.2 QoS statistics
Version 1.8	02.03.2021	Synchronization with version 4.0.15.3
Version 1.7	10.02.2021	Sections changed: 6.1 Ethernet interface parameters
Version 1.6	28.11.2020	Sections changed: 10.2 Spanning-tree protocol

Version 1.5	27.10.2020	Sections added: 18 Configuring protection against DoS attacks
Version 1.4	16.10.2020	Synchronization with version 4.0.14.2
Version 1.3	14.09.2020	Sections changed: 16.3 IP-source Guard
Version 1.2	19.02.2020	Sections added: 4.3 Stack parameters 8 Configuring IPv6 addressing 15 Power over Ethernet (PoE)  Sections changed: 1 SNMP server and SNMP-TRAP sending configuration 2 Short descriptions 4.1 System resources 4.2 System parameters 4.4 Device management 5 System time configuration 6.1 Ethernet interface parameters 6.3 Errdisable state configuration and monitoring 10.2 Spanning-tree protocol 12.1 AAA mechanism 12.2 Access configuration 13 Port Mirroring 16.6 Loopback detection mechanism
Version 1.1	13.07.2018	First issue
<b>Firmware Version</b>	<b>4.0.22</b>	

CONTENTS

1	SNMP SERVER AND SNMP-TRAP SENDING CONFIGURATION .....	7
2	SHORT DESCRIPTIONS .....	7
3	FILE OPERATIONS .....	10
3.1	Saving the configuration .....	10
3.2	Operation with TFTP server .....	11
3.3	Operation with SFTP and SCP servers .....	13
3.4	Downloading a license file to a device .....	16
3.5	Switch autoconfiguration .....	17
3.6	Firmware update .....	17
4	SYSTEM MANAGEMENT .....	21
4.1	System resources .....	21
4.2	System parameters .....	31
4.3	Stack parameters .....	34
4.4	Device management .....	35
5	SYSTEM TIME CONFIGURATION .....	39
6	INTERFACE CONFIGURATION .....	41
6.1	Ethernet interface parameters .....	41
6.2	Configuring VLAN .....	51
6.3	Errdisable state configuration and monitoring .....	56
6.4	Configuring voice vlan .....	58
6.5	Configuring LLDP .....	59
6.6	Configuring Flex-link .....	60
7	CONFIGURING IPV4 ADDRESSING .....	62
8	CONFIGURING IPV6 ADDRESSING .....	64
9	CONFIGURING GREEN ETHERNET .....	65
10	CONFIGURING RING PROTOCOLS .....	66
10.1	ERPS protocol .....	66
10.2	Spanning-tree protocol .....	68
11	MULTICAST ADDRESSING .....	73
11.1	Multicast addressing rules .....	73
11.2	Multicast traffic restriction functions .....	75
12	CONTROL FUNCTIONS .....	78
12.1	AAA mechanism .....	78
12.2	Access configuration .....	82
13	PORT MIRRORING .....	84
14	PHYSICAL LAYER DIAGNOSTIC FUNCTIONS .....	87
14.1	Copper-wire cable diagnostics .....	87
14.2	Optical transceiver diagnostics .....	89
15	POWER OVER ETHERNET (POE) .....	91
16	SECURITY FUNCTIONS .....	94
16.1	Port security functions .....	94
16.2	DHCP control and option 82 .....	98
16.3	IP-source Guard .....	101
16.4	ARP Inspection .....	102
16.5	Port-based client authentication (802.1x) .....	104
16.6	Loopback detection mechanism .....	108
16.7	Broadcast storm control .....	110
17	CONFIGURING IP AND MAC ACL .....	112
18	CONFIGURING PROTECTION AGAINST DOS ATTACKS .....	117
19	QUALITY OF SERVICE (QOS) .....	118
19.1	QoS configuration .....	118
19.2	QoS statistics .....	121

---

20	ROUTING .....	121
20.1	Static routing .....	121
20.2	Dynamic routing .....	122
APPENDIX A. BIT MASK CALCULATION METHOD.....		123
APPENDIX B: EXAMPLE OF CREATING A STANDARD IP ACL.....		124
APPENDIX C. EXAMPLE OF CREATING, FILLING AND REMOVING AN OFFSET-LIST WITH MAC ACL.....		131

## DOCUMENT CONVENTIONS

Typographic element	Description
[ ]	Square brackets are used to indicate optional parameters in the command line; when entered, they provide additional options.
{ }	In the command line, mandatory parameters are shown in curly braces.
«,» «-»	In the command description, these characters are used to specify ranges.
«   »	In the command description, this character means 'or'.
« / »	This sign separates possible and default values when specifying variable values.
<i>Calibri Italic</i>	Calibri Italic is used to indicate variables and parameters that should be replaced with an appropriate word or string.
<b><i>Bold italic</i></b>	Notes and warnings are shown in bold italic.
< <b><i>Bold Italic</i></b> >	Keyboard keys are shown in bold italic within angle brackets.
<b>Courier New</b>	Command examples are shown in Courier New Bold.

### Notes and Warnings



Notes contain important information, tips, or recommendations on device operation and configuration.



Warnings inform the user about situations that may be harmful to the user, cause damage to the device, malfunction or data loss.

## 1 SNMP SERVER AND SNMP-TRAP SENDING CONFIGURATION

```
snmp-server server
snmp-server community public ro
snmp-server community private rw
snmp-server host 192.168.1.1 traps version 2c private
```

## 2 SHORT DESCRIPTIONS

- **ifIndex** — port index;

May take the following values:

1. Access switches

Switch model	Indexes
MES2308 MES2308R MES2308P	- indexes 49-96 — gigabitethernet 1/0/1-48; - indexes 157-204 — gigabitethernet 2/0/1-48; - indexes 256-303 — gigabitethernet 3/0/1-48; - indexes 373-420 — gigabitethernet 4/0/1-48; - indexes 481-528 — gigabitethernet 5/0/1-48; - indexes 589-636 — gigabitethernet 6/0/1-48; - indexes 697-744 — gigabitethernet 7/0/1-48;
MES2324 MES2324B MES2324F MES2324FB MES2348 MES2348B MES2324P	- indexes 805-852 — gigabitethernet 8/0/1-48; - indexes 105-108 — tengigabitethernet 1/0/1-4; - indexes 213-216 — tengigabitethernet 2/0/1-4; - indexes 321-324 — tengigabitethernet 3/0/1-4; - indexes 429-432 — tengigabitethernet 4/0/1-4; - indexes 537-540 — tengigabitethernet 5/0/1-4;
MES2348P	- indexes 645-648 — tengigabitethernet 6/0/1-4; - indexes 753-756 — tengigabitethernet 7/0/1-4; - indexes 861-864 — tengigabitethernet 8/0/1-4; - indexes 1000-1047 — Port-Channel 1/0/1-48; - indexes 100000-104095 — VLAN 1-4096.

2. Aggregation switches

Switch model	Indexes
MES3324 MES3324F MES3308F MES3316F MES3348 MES3348F	- indexes 49-96 — gigabitethernet 1/0/1-48; - indexes 157-204 — gigabitethernet 2/0/1-48; - indexes 256-303 — gigabitethernet 3/0/1-48; - indexes 373-420 — gigabitethernet 4/0/1-48; - indexes 481-528 — gigabitethernet 5/0/1-48;

	<ul style="list-style-type: none"> <li>- indexes 589-636 — gigabitethernet 6/0/1-48;</li> <li>- indexes 697-744 — gigabitethernet 7/0/1-48;</li> <li>- indexes 805-852 — gigabitethernet 8/0/1-48;</li> <li>- indexes 105-108 — tengigabitethernet 1/0/1-4;</li> <li>- indexes 105-108 — tengigabitethernet 1/0/1-4;</li> <li>- indexes 213-216 — tengigabitethernet 2/0/1-4;</li> <li>- indexes 321-324 — tengigabitethernet 3/0/1-4;</li> <li>- indexes 429-432 — tengigabitethernet 4/0/1-4;</li> <li>- indexes 537-540 — tengigabitethernet 5/0/1-4;</li> <li>- indexes 645-648 — tengigabitethernet 6/0/1-4;</li> <li>- indexes 753-756 — tengigabitethernet 7/0/1-4;</li> <li>- indexes 861-864 — tengigabitethernet 8/0/1-4;</li> <li>- indexes 1000-1047 — Port-Channel 1/0/1-48;</li> <li>- indexes 100000-104095 — VLAN 1-4096.</li> </ul>
--	--

### 3. Industrial switches

Switch model	Indexes
MES2328I MES3508 MES3508P MES3510P	<ul style="list-style-type: none"> <li>- indexes 49-76 - gigabitethernet 1/0/1-28;</li> <li>- indexes 157-184 - gigabitethernet 2/0/1-28;</li> <li>- indexes 256-283 - gigabitethernet 3/0/1-28;</li> <li>- indexes 373-400 - gigabitethernet 4/0/1-28;</li> <li>- indexes 481-508 - gigabitethernet 5/0/1-28;</li> <li>- indexes 589-616 - gigabitethernet 6/0/1-28;</li> <li>- indexes 697-724 - gigabitethernet 7/0/1-28;</li> <li>- indexes 805-832 - gigabitethernet 8/0/1-28;</li> <li>- indexes 1000-1047 — Port-Channel 1/0/1-48;</li> <li>- indexes 100000-104095 — VLAN 1-4096.</li> </ul>

### 4. Data center switches

Switch model	Indexes
MES5324	<ul style="list-style-type: none"> <li>- indexes 1-24 — tengigabitethernet 1/0/1-24;</li> <li>- indexes 53-76 — tengigabitethernet 2/0/1-24;</li> <li>- indexes 105-128 — tengigabitethernet 3/0/1-24;</li> <li>- indexes 157-180 — tengigabitethernet 4/0/1-24;</li> <li>- indexes 209-232 — tengigabitethernet 5/0/1-24;</li> <li>- indexes 261-284 — tengigabitethernet 6/0/1-24;</li> <li>- indexes 313-336 — tengigabitethernet 7/0/1-24;</li> <li>- indexes 365-388 — tengigabitethernet 8/0/1-24;</li> <li>- indexes 25-28 — fortygigabitethernet 1/0/1-4;</li> </ul>



	<ul style="list-style-type: none"> <li>- indexes 77-80 — fortygigabitethernet2/0/1-4;</li> <li>- indexes 129-132 — fortygigabitethernet3/0/1-4;</li> <li>- indexes 181-184 — fortygigabitethernet4/0/1-4;</li> <li>- indexes 233-236 — fortygigabitethernet5/0/1-4;</li> <li>- indexes 285-288 — fortygigabitethernet6/0/1-4;</li> <li>- indexes 337-340 — fortygigabitethernet7/0/1-4;</li> <li>- indexes 389-392 — fortygigabitethernet8/0/1-4;</li> <li>- indexes 1000-1047 — Port-Channel 1/0/1-48;</li> <li>- indexes 100000-104095 — VLAN 1-4096.</li> </ul>
--	--

- **index-of-rule** — rule index in ACL. Always a multiple of 20. If the indexes are not divisible by 20 when the rules are created, the sequence numbers of the rules in the ACL will be divisible by 20 after the switch is rebooted;
- **The value of field N** — in IP and MAC ACL any rule occupies from one to three fields depending on its structure;
- **IP address** — IP address for switch management;

In the examples given in the document the following IP address is used for management: **192.168.1.30**;

- **ip address of tftp server** — TFTP server IP address;

In the examples given in the document the following TFTP server IP address is used: **192.168.1.1**;

- **community** — community string (password) for the access via SNMP.

In the examples given in the document the following *community* are used:

**private** — rights for writing (rw);

**public** — rights for reading (ro).

## 3 FILE OPERATIONS

### 3.1 Saving the configuration

#### Saving the configuration to non-volatile memory

**MIB:** rlcopy.mib

**Tables used:** rlCopyEntry — 1.3.6.1.4.1.89.87.2.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.87.2.1.3.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.7.1 i {runningConfig(2)} \
  1.3.6.1.4.1.89.87.2.1.8.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.12.1 i {startupConfig (3)} \
  1.3.6.1.4.1.89.87.2.1.17.1 i {createAndGo (4)}
```

#### **Example of saving to non-volatile memory**

CLI command:

```
copy running-config startup-config
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.87.2.1.3.1 i 1 \
  1.3.6.1.4.1.89.87.2.1.7.1 i 2 \
  1.3.6.1.4.1.89.87.2.1.8.1 i 1 \
  1.3.6.1.4.1.89.87.2.1.12.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.17.1 i 4
```

#### Saving the configuration to volatile memory from non-volatile one

**MIB:** rlcopy.mib

**Tables used:** rlCopyEntry — 1.3.6.1.4.1.89.87.2.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.87.2.1.3.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.7.1 i {startupConfig (3)} \
  1.3.6.1.4.1.89.87.2.1.8.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.12.1 i {runningConfig(2)} \
  1.3.6.1.4.1.89.87.2.1.17.1 i {createAndGo (4)}
```

#### **Example of saving to volatile memory**

CLI command:

```
copy startup-config running-config
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.87.2.1.3.1 i 1 \
  1.3.6.1.4.1.89.87.2.1.7.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.8.1 i 1 \
  1.3.6.1.4.1.89.87.2.1.12.1 i 2 \
  1.3.6.1.4.1.89.87.2.1.17.1 i 4
```

### Deleting the configuration from non-volatile memory

**MIB:** rlmng.mib

**Tables used:** rndAction — 1.3.6.1.4.1.89.1.2

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.1.2.0 i {eraseStartupCDB (20)}
```

#### **Startup-config deletion example**

CLI command:

```
delete startup-config
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.1.2.0 i 20
```

## **3.2 Operation with TFTP server**

### Copying the configuration from volatile memory to a TFTP server

**MIB:** rlcopymib

**Tables used:** rlcopEntry — 1.3.6.1.4.1.89.87.2.1

```
snmpset -v2c -c <community> -t 5 -r 3 <IP address> \
  1.3.6.1.4.1.89.87.2.1.3.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.7.1 i {runningConfig(2)} \
  1.3.6.1.4.1.89.87.2.1.8.1 i {tftp(3)} \
  1.3.6.1.4.1.89.87.2.1.9.1 a {ip address of tftp server} \
  1.3.6.1.4.1.89.87.2.1.11.1 s "MES-config.cfg" \
  1.3.6.1.4.1.89.87.2.1.17.1 i {createAndGo (4)}
```

#### **Example of copying from running-config to a TFTP server**

CLI command:

```
copy running-config tftp://192.168.1.1/MES-config.cfg
```

SNMP command:

```
snmpset -v2c -c private -t 5 -r 3 192.168.1.30 \
  1.3.6.1.4.1.89.87.2.1.3.1 i 1 \
  1.3.6.1.4.1.89.87.2.1.7.1 i 2 \
  1.3.6.1.4.1.89.87.2.1.8.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.9.1 a 192.168.1.1 \
  1.3.6.1.4.1.89.87.2.1.11.1 s "MES-config.cfg" \
  1.3.6.1.4.1.89.87.2.1.17.1 i 4
```

### Copying the configuration to volatile memory from a TFTP server

**MIB:** rcopy.mib

**Tables used:** rICopyEntry — 1.3.6.1.4.1.89.87.2.1

```
snmpset -v2c -c <community> -t 5 -r 3 <IP address> \
  1.3.6.1.4.1.89.87.2.1.3.1 i {tftp(3)} \
  1.3.6.1.4.1.89.87.2.1.4.1 a {ip address of tftp server} \
  1.3.6.1.4.1.89.87.2.1.6.1 s "MES-config.cfg" \
  1.3.6.1.4.1.89.87.2.1.8.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.12.1 i {runningConfig(2)} \
  1.3.6.1.4.1.89.87.2.1.17.1 i {createAndGo (4)}
```

#### **Example of copying from a TFTP server to running-config**

**CLI command:**

```
copy tftp://192.168.1.1/MES-config.cfg running-config
```

**SNMP command:**

```
snmpset -v2c -c private -t 5 -r 3 192.168.1.30 \
  1.3.6.1.4.1.89.87.2.1.3.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.4.1 a 192.168.1.1 \
  1.3.6.1.4.1.89.87.2.1.6.1 s "MES-config.cfg" \
  1.3.6.1.4.1.89.87.2.1.8.1 i 1 \
  1.3.6.1.4.1.89.87.2.1.12.1 i 2 \
  1.3.6.1.4.1.89.87.2.1.17.1 i 4
```

### Copying the configuration from non-volatile memory to a TFTP server

**MIB:** rcopy.mib

**Tables used:** rICopyEntry — 1.3.6.1.4.1.89.87.2.1

```
snmpset -v2c -c <community> -t 5 -r 3 <IP address> \
  1.3.6.1.4.1.89.87.2.1.3.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.7.1 i {startupConfig (3)} \
  1.3.6.1.4.1.89.87.2.1.8.1 i {tftp(3)} \
  1.3.6.1.4.1.89.87.2.1.9.1 a {ip address of tftp server} \
  1.3.6.1.4.1.89.87.2.1.11.1 s "MES-config.cfg" \
  1.3.6.1.4.1.89.87.2.1.17.1 i {createAndGo (4)}
```

#### **Example of copying from startup-config to a TFTP server**

**CLI command:**

```
copy startup-config tftp://192.168.1.1/MES-config.cfg
```

**SNMP command:**

```
snmpset -v2c -c private -t 5 -r 3 192.168.1.30 \
  1.3.6.1.4.1.89.87.2.1.3.1 i 1 \
  1.3.6.1.4.1.89.87.2.1.7.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.8.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.9.1 a 192.168.1.1 \
  1.3.6.1.4.1.89.87.2.1.11.1 s "MES-config.cfg" \
  1.3.6.1.4.1.89.87.2.1.17.1 i 4
```

### Copying the configuration to non-volatile memory from a TFTP server

**MIB:** rlcopymib

**Tables used:** rICopyEntry — 1.3.6.1.4.1.89.87.2.1

```
snmpset -v2c -c <community> -t 5 -r 3 <IP address> \
  1.3.6.1.4.1.89.87.2.1.3.1 i {tftp(3)} \
  1.3.6.1.4.1.89.87.2.1.4.1 a {ip address of tftp server} \
  1.3.6.1.4.1.89.87.2.1.6.1 s "MES-config.cfg" \
  1.3.6.1.4.1.89.87.2.1.8.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.12.1 i {startupConfig (3)} \
  1.3.6.1.4.1.89.87.2.1.17.1 i {createAndGo (4)}
```

#### **Example of copying startup-config from a TFTP server**

**CLI command:**

```
boot config tftp://192.168.1.1/MES-config.cfg
```

**SNMP command:**

```
snmpset -v2c -c private -t 5 -r 3 192.168.1.30 \
  1.3.6.1.4.1.89.87.2.1.3.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.4.1 a 192.168.1.1 \
  1.3.6.1.4.1.89.87.2.1.6.1 s "MES-config.cfg" \
  1.3.6.1.4.1.89.87.2.1.8.1 i 1 \
  1.3.6.1.4.1.89.87.2.1.12.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.17.1 i 4
```

## **3.3 Operation with SFTP and SCP servers**

### Copying the configuration from volatile memory to an SFTP or SCP server

**MIB:** rlcopymib

**Tables used:** rICopyInetTable — 1.3.6.1.4.1.89.87.8

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.87.8.1.3.1 i {local(1)} \
  1.3.6.1.4.1.89.87.8.1.7.1 s {"running-config"} \
  1.3.6.1.4.1.89.87.8.1.8.1 i {runningConfig(2)} \
  1.3.6.1.4.1.89.87.8.1.9.1 i {sftp(11)} \
  1.3.6.1.4.1.89.87.8.1.10.1 i {ipv4(1)} \
  1.3.6.1.4.1.89.87.8.1.11.1 x {"IP address in HEX-format"} \
  1.3.6.1.4.1.89.87.8.1.13.1 s {"MES-config.cfg"} \
  1.3.6.1.4.1.89.87.8.1.14.1 I {other(1)} \
  1.3.6.1.4.1.89.87.8.1.19.1 i {createAndGo(4)} \
  1.3.6.1.4.1.89.87.8.1.23.1 s {"username:password@"}
```

#### **Example of copying from running-config to an SFTP server**

**CLI command:**

```
copy running-config sftp://username:password@192.168.1.1/MES-config.cfg
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.87.8.1.3.1 i 1 \
  1.3.6.1.4.1.89.87.8.1.7.1 s "running-config" \
  1.3.6.1.4.1.89.87.8.1.8.1 i 2 \
  1.3.6.1.4.1.89.87.8.1.9.1 i 11 \
  1.3.6.1.4.1.89.87.8.1.10.1 i 1 \
```

```

1.3.6.1.4.1.89.87.8.1.11.1 x "0xC0A80101" \
1.3.6.1.4.1.89.87.8.1.13.1 s "MES-config.cfg" \
1.3.6.1.4.1.89.87.8.1.14.1 i 1 \
1.3.6.1.4.1.89.87.8.1.19.1 i 4 \
1.3.6.1.4.1.89.87.8.1.23.1 s "username:password@"

```

### Copying the configuration to volatile memory from an SFTP or SCP server

**MIB:** rlcopy.mib

**Tables used:** rlCopyInetTable — 1.3.6.1.4.1.89.87.8

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.87.8.1.3.1 i {sftp(11), scp(5)} \
1.3.6.1.4.1.89.87.8.1.4.1 i {ipv4(1)} \
1.3.6.1.4.1.89.87.8.1.5.1 x {"IP address in HEX-format"} \
1.3.6.1.4.1.89.87.8.1.7.1 s {"MES-config.cfg"} \
1.3.6.1.4.1.89.87.8.1.8.1 i {startupConfig(3)} \
1.3.6.1.4.1.89.87.8.1.9.1 i {local(1)} \
1.3.6.1.4.1.89.87.8.1.13.1 s {"running-config"} \
1.3.6.1.4.1.89.87.8.1.14.1 i {runningConfig(2)} \
1.3.6.1.4.1.89.87.8.1.19.1 i {createAndGo(4)} \
1.3.6.1.4.1.89.87.8.1.23.1 s {"username:password@"}

```

### **Example of copying from an SFTP server to running-config**

**CLI command:**

```
copy sftp://username:password@192.168.1.1/MES-config.cfg running-config
```

**SNMP command:**

```

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.87.8.1.3.1 i 11 \
1.3.6.1.4.1.89.87.8.1.4.1 i 1 \
1.3.6.1.4.1.89.87.8.1.5.1 x "0xC0A80101" \
1.3.6.1.4.1.89.87.8.1.7.1 s "MES-config.cfg" \
1.3.6.1.4.1.89.87.8.1.8.1 i 3 \
1.3.6.1.4.1.89.87.8.1.9.1 i 1 \
1.3.6.1.4.1.89.87.8.1.13.1 s "running-config" \
1.3.6.1.4.1.89.87.8.1.14.1 i 2 \
1.3.6.1.4.1.89.87.8.1.19.1 i 4 \
1.3.6.1.4.1.89.87.8.1.23.1 s "username:password@"

```

### Copying the configuration from non-volatile memory to an SFTP or SCP server

**MIB:** rlcopy.mib

**Tables used:** rlCopyInetTable — 1.3.6.1.4.1.89.87.8

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.87.8.1.3.1 i {local(1)} \
1.3.6.1.4.1.89.87.8.1.7.1 s {"startup-config"} \
1.3.6.1.4.1.89.87.8.1.8.1 i {startupConfig(3)} \
1.3.6.1.4.1.89.87.8.1.9.1 i {sftp(11), scp(5)} \
1.3.6.1.4.1.89.87.8.1.10.1 i {ipv4(1)} \
1.3.6.1.4.1.89.87.8.1.11.1 x {"IP address in HEX-format"} \
1.3.6.1.4.1.89.87.8.1.13.1 s {"MES-startup-config.cfg"} \
1.3.6.1.4.1.89.87.8.1.14.1 i {other(1)} \
1.3.6.1.4.1.89.87.8.1.19.1 i {createAndGo(4)} \
1.3.6.1.4.1.89.87.8.1.23.1 s {"username:password@"}

```

### Example of copying from startup-config to an SFTP server

#### CLI command:

```
sftp://username:password@192.168.1.1/MES-startup-config.cfg running-config
```

#### SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.87.8.1.3.1 i 1 \
1.3.6.1.4.1.89.87.8.1.7.1 s "startup-config" \
1.3.6.1.4.1.89.87.8.1.8.1 i 3 \
1.3.6.1.4.1.89.87.8.1.9.1 i 11 \
1.3.6.1.4.1.89.87.8.1.10.1 i 1 \
1.3.6.1.4.1.89.87.8.1.11.1 x "0xC0A80101" \
1.3.6.1.4.1.89.87.8.1.13.1 s "MES-startup-config.cfg" \
1.3.6.1.4.1.89.87.8.1.14.1 i 1 \
1.3.6.1.4.1.89.87.8.1.19.1 i 4 \
1.3.6.1.4.1.89.87.8.1.23.1 s "username:password@"
```

### Copying the configuration to non-volatile memory from an SFTP or SCP server

**MIB:** rlcopy.mib

**Tables used:** rICopyInetTable — 1.3.6.1.4.1.89.87.8

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.87.8.1.3.1 i {sftp(11), scp(5)} \
1.3.6.1.4.1.89.87.8.1.4.1 i {ipv4(1)} \
1.3.6.1.4.1.89.87.8.1.5.1 x {"IP address in HEX-format"} \
1.3.6.1.4.1.89.87.8.1.7.1 s {"MES-config.cfg"} \
1.3.6.1.4.1.89.87.8.1.8.1 i {other(1)} \
1.3.6.1.4.1.89.87.8.1.9.1 i {local(1)} \
1.3.6.1.4.1.89.87.8.1.13.1 s {"startup-config"} \
1.3.6.1.4.1.89.87.8.1.14.1 i {startupConfig(3)} \
1.3.6.1.4.1.89.87.8.1.19.1 i {createAndGo(4)} \
1.3.6.1.4.1.89.87.8.1.23.1 s {"username:password@"}
```

### Example of copying from an SFTP server to startup-config

#### SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.87.8.1.3.1 i 11 \
1.3.6.1.4.1.89.87.8.1.4.1 i 1 \
1.3.6.1.4.1.89.87.8.1.5.1 x "0xC0A80101" \
1.3.6.1.4.1.89.87.8.1.7.1 s "MES-startup-config.cfg" \
1.3.6.1.4.1.89.87.8.1.8.1 i 1 \
1.3.6.1.4.1.89.87.8.1.9.1 i 1 \
1.3.6.1.4.1.89.87.8.1.13.1 s "startup-config" \
1.3.6.1.4.1.89.87.8.1.14.1 i 3 \
1.3.6.1.4.1.89.87.8.1.19.1 i 4 \
1.3.6.1.4.1.89.87.8.1.23.1 s "username:password@"
```

Also, copying to non-volatile memory from an SFTP or SCP server can be implemented via boot config:

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.87.8.1.3.1 i {sftp(11), scp(5)} \
1.3.6.1.4.1.89.87.8.1.4.1 i {ipv4(1)} \
1.3.6.1.4.1.89.87.8.1.5.1 x {"IP address in HEX-format"} \
1.3.6.1.4.1.89.87.8.1.7.1 s {"MES-config.cfg"} \
1.3.6.1.4.1.89.87.8.1.8.1 i {other(1)} \
1.3.6.1.4.1.89.87.8.1.9.1 i {local(1)} \
1.3.6.1.4.1.89.87.8.1.14.1 i {startupConfig(3)} \
```

```
1.3.6.1.4.1.89.87.8.1.19.1 i {createAndGo(4)} \
1.3.6.1.4.1.89.87.8.1.23.1 s {"username:password@"}
```

### Example of copying from an SFTP server to startup-config

#### CLI command:

```
boot config sftp://username:password@192.168.1.1/MES-startup-config.cfg
```

#### SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.87.8.1.3.1 i 11 \
1.3.6.1.4.1.89.87.8.1.4.1 i 1 \
1.3.6.1.4.1.89.87.8.1.5.1 x "0xC0A80101" \
1.3.6.1.4.1.89.87.8.1.7.1 s "MES-startup-config.cfg" \
1.3.6.1.4.1.89.87.8.1.8.1 i 1 \
1.3.6.1.4.1.89.87.8.1.9.1 i 1 \
1.3.6.1.4.1.89.87.8.1.14.1 i 3 \
1.3.6.1.4.1.89.87.8.1.19.1 i 4 \
1.3.6.1.4.1.89.87.8.1.23.1 s "username:password@"
```

## 3.4 Downloading a license file to a device

### Downloading a license file to a device from a TFTP, SFTP, or SCP server

**MIB:** rlcopy.mib

**Tables used:** rlCopyInetTable — 1.3.6.1.4.1.89.87.8

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.87.8.1.3.1 i {sftp(11), scp(5), tftp(3)} \
1.3.6.1.4.1.89.87.8.1.4.1 i {ipv4(1)} \
1.3.6.1.4.1.89.87.8.1.5.1 x {"IP address in HEX-format"} \
1.3.6.1.4.1.89.87.8.1.7.1 s {"Licence name"} \
1.3.6.1.4.1.89.87.8.1.8.1 i {other(1)} \
1.3.6.1.4.1.89.87.8.1.9.1 i {local(1)} \
1.3.6.1.4.1.89.87.8.1.14.1 i {licence(14)} \
1.3.6.1.4.1.89.87.8.1.19.1 i {createAndGo(4)} \
1.3.6.1.4.1.89.87.8.1.23.1 s {"username:password@"}
```

### Example of copying a license from an SFTP server

#### CLI command:

```
boot license sftp://username:password@192.168.1.1/license
```

#### SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.87.8.1.3.1 i 11 \
1.3.6.1.4.1.89.87.8.1.4.1 i 1 \
1.3.6.1.4.1.89.87.8.1.5.1 x "0xC0A80101" \
1.3.6.1.4.1.89.87.8.1.7.1 s "MES-Licence" \
1.3.6.1.4.1.89.87.8.1.8.1 i 1 \
1.3.6.1.4.1.89.87.8.1.9.1 i 1 \
1.3.6.1.4.1.89.87.8.1.14.1 i 14 \
1.3.6.1.4.1.89.87.8.1.19.1 i 4 \
1.3.6.1.4.1.89.87.8.1.23.1 s "username:password@"
```



### 3.5 Switch autoconfiguration

#### Enabling DHCP-based autoconfiguration (enabled by default)

**MIB:** radlan-dhcpcl-mib.mib

**Tables used:** rIDhcpCLOption67Enable — 1.3.6.1.4.1.89.76.9

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.76.9.0 i {enable(1), disable(2)}
```

#### **Example of enabling autoconfiguration based on DHCP**

**CLI command:**

```
boot host auto-config
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.76.9.0 i 1
```

### 3.6 Firmware update

#### Switch firmware update

**Performed in two steps:**

##### 1. Downloading a firmware image from a TFTP server

**MIB:** rIcopy.mib

**Tables used:** rICopyEntry — 1.3.6.1.4.1.89.87.2.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.87.2.1.3.1 i {tftp (3)} \
  1.3.6.1.4.1.89.87.2.1.4.1 a {ip add of tftp server} \
  1.3.6.1.4.1.89.87.2.1.6.1 s "image name" \
  1.3.6.1.4.1.89.87.2.1.8.1 i {local(1)} \
  1.3.6.1.4.1.89.87.2.1.12.1 i {image(8)} \
  1.3.6.1.4.1.89.87.2.1.17.1 i {createAndGo(4)}
```

#### **Example of viewing a firmware image**

**CLI command:**

```
boot system tftp://192.168.1.1/mes3300-409-R478.ros
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.87.2.1.3.1 i 3 \
  1.3.6.1.4.1.89.87.2.1.4.1 a 192.168.1.1 \
  1.3.6.1.4.1.89.87.2.1.6.1 s "mes3300-409-R478.ros" \
  1.3.6.1.4.1.89.87.2.1.8.1 i 1 1.3.6.1.4.1.89.87.2.1.12.1 i 8 \
  1.3.6.1.4.1.89.87.2.1.17.1 i 4
```

## 2. Downloading a firmware image from an SFTP or SCP server

**MIB:** rlcopy.mib

**Tables used:** rlCopyInetTable — 1.3.6.1.4.1.89.87.8

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.87.8.1.3.1 i {sftp(11), scp(5)} \
1.3.6.1.4.1.89.87.8.1.4.1 i {ipv4(1)} \
1.3.6.1.4.1.89.87.8.1.5.1 x {"IP address in HEX-format"} \
1.3.6.1.4.1.89.87.8.1.7.1 s {"image name"} \
1.3.6.1.4.1.89.87.8.1.8.1 i {other(1)} \
1.3.6.1.4.1.89.87.8.1.9.1 i {local(1)} \
1.3.6.1.4.1.89.87.8.1.14.1 i {image(8)} \
1.3.6.1.4.1.89.87.8.1.19.1 i {createAndGo(4)} \
1.3.6.1.4.1.89.87.8.1.23.1 s {"username:password@"}
```

### Example of viewing a firmware image

CLI command:

```
boot system sftp://username:password@192.168.1.1/mes3300-4021-R3.ros
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.87.8.1.3.1 i 11 \
1.3.6.1.4.1.89.87.8.1.4.1 i 1 \
1.3.6.1.4.1.89.87.8.1.5.1 x "0xC0A80101" \
1.3.6.1.4.1.89.87.8.1.7.1 s "mes3300-4021-R3.ros" \
1.3.6.1.4.1.89.87.8.1.8.1 i 1 \
1.3.6.1.4.1.89.87.8.1.9.1 i 1 \
1.3.6.1.4.1.89.87.8.1.14.1 i 8 \
1.3.6.1.4.1.89.87.8.1.19.1 i 4 \
1.3.6.1.4.1.89.87.8.1.23.1 s "name:password@"
```

## 3. Active switch image change

**MIB:** radlan-deviceparams-mib.mib

**Tables used:** rndActiveSoftwareFileAfterReset — 1.3.6.1.4.1.89.2.13.1.1.3

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.2.13.1.1.3.1 i {image1 (1), image2 (2)}
```

### Example of active switch image change

CLI command:

```
boot system inactive-image
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.2.13.1.1.3.1 i 1
```



**The command is applied automatically after the firmware is downloaded from a TFTP, SFTP, or SCP server.**

### Switch reboot

**MIB:** rlmng.mib

**Tables used:** rlRebootDelay — 1.3.6.1.4.1.89.1.10

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.1.10.0 t {time delay before rebooting}
```

#### **Example of a reboot delayed by 8 minutes**

CLI command:

```
reload in 8
```

SNMP command:

```
snmpset -v2c -c private -r 0 192.168.1.30 \
1.3.6.1.4.1.89.1.10.0 t 48000
```



**To reboot immediately, the value t=0 is required.**

### Viewing the firmware image

**MIB:** radlan-deviceparams-mib.mib

**Tables used:** rndActiveSoftwareFile — 1.3.6.1.4.1.89.2.13.1.1.2

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.2.13.1.1.2
```

#### **Example of viewing a firmware image**

CLI command:

```
show bootvar
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.2.13.1.1.2
```



**Possible options:**

**image1(1)**

**image2(2)**

**rndActiveSoftwareFileAfterReset — 1.3.6.1.4.1.89.2.13.1.1.3 — here you can view the active image after the firmware reboot.**

### Viewing uploaded firmware images

**MIB:** radlan-deviceparams-mib.mib

**Tables used:** rndImageInfoTable — 1.3.6.1.4.1.89.2.16.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.2.16.1
```

### Example of viewing downloaded firmware images

CLI command:

```
show bootvar
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.2.16.1
```

### Viewing the current switch firmware version

**MIB:** radlan-deviceparams-mib.mib

**Tables used:** rndBrgVersion — 1.3.6.1.4.1.89.2.4

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.2.4
```

### Example of viewing the current switch firmware version

CLI command:

```
show version
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.2.4
```

### Viewing the current hardware (HW) version

**MIB:** radlan-deviceparams-mib.mib

**Tables used:** genGroupHWVersion — 1.3.6.1.4.1.89.2.11.1

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.2.11.1
```

### Example of viewing the current firmware version

CLI command:

```
show system id
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.2.11.1
```

## 4 SYSTEM MANAGEMENT

### 4.1 System resources

#### Viewing switch serial number

**MIB:** rphysdescription.mib

**Tables used:** rIPhdUnitGenParamSerialNum — 1.3.6.1.4.1.89.53.14.1.5

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.53.14.1.5
```

#### **Example of viewing the serial port of the switch**

CLI command:  
show system id

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.53.14.1.5

#### Viewing information on tcam utilization

**MIB:** RADLAN-QOS-CLI-MIB

**Tables used:** rIQosClassifierUtilizationPercent — 1.3.6.1.4.1.89.88.36.1.1.2

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.36.1.1.2
```

#### **Example of viewing information about tcam utilization**

CLI command:  
show system tcam utilization

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.88.36.1.1.2

#### Viewing the maximum number of hosts

**MIB:** rltuning.mib

**Tables used:** rsMaxIpSfftEntries — 1.3.6.1.4.1.89.29.8.9.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.29.8.9.1
```

#### **Example of viewing the maximum number of hosts**

CLI command:  
show system router resources

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.29.8.9.1

### Viewing the number of hosts used

**MIB:** rlfft.mib

**Tables used:** rlSysmgTcamAllocInUseEntries — 1.3.6.1.4.1.89.204.1.1.1.5

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.204.1.1.1.5.5.116.99.97.109.49.1
```

#### **Example of viewing the number of hosts used**

CLI command:

```
show system router resources
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.204.1.1.1.5.5.116.99.97.109.49.1
```

### Viewing the maximum number of routes

**MIB:** rltuning.mib

**Tables used:** rsMaxIpPrefixes — 1.3.6.1.4.1.89.29.8.21.1

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.29.8.21.1
```

#### **Example of viewing the maximum number of routes**

CLI command:

```
show system router resources
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.29.8.21.1
```

### Viewing the number of routes used

**MIB:** rlip.mib

**Tables used:** rllpTotalPrefixesNumber — 1.3.6.1.4.1.89.26.25

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.26.25
```

#### **Example of viewing the number of routes used**

CLI command:

```
show system router resources
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.26.25
```

### Viewing the maximum number of IP interfaces

**MIB:** rltuning.mib

**Tables used:** rsMaxIpInterfaces — 1.3.6.1.4.1.89.29.8.25.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.29.8.25.1
```

#### **Example of viewing the maximum number of IP interfaces**

CLI command:  
show system router resources

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.29.8.25.1

### Viewing the number of IP interfaces used

**MIB:** rlip.mib

**Tables used:** rIpAddressesNumber — 1.3.6.1.4.1.89.26.23

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.26.23
```

#### **Example of viewing the number of IP interfaces used**

CLI command:  
show system router resources

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.26.23

### Viewing the switch's system MAC address

**MIB:** rphysdescription.mib

**Tables used:** rIPhdStackMacAddr — 1.3.6.1.4.1.89.53.4.1.7

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.53.4.1.7
```

#### **Example of viewing the switch's system MAC address**

CLI command:  
show system

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.53.4.1.7

### Viewing switch Uptime



The command is relevant for Uptime less than 497 days.

**MIB:** SNMPv2-MIB

**Tables used:** sysUpTime — 1.3.6.1.2.1.1.3

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.1.3
```

#### **Example of viewing the switch Uptime**

CLI command:

```
show system
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.1.3
```

### Viewing the switch's Uptime counter in seconds



The team is relevant for Uptime less than 136 years.

**MIB:** eltSystemGlobal.mib

**Tables used:** eltsysUpTimeInSec — 1.3.6.1.4.1.35265.1.23.1.10.3.1.1

```
snmpget -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.1.23.1.10.3.1.1.0
```

#### **Example of viewing the switch's Uptime counter in seconds**

SNMP command:

```
snmpget -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.1.10.3.1.1.0
```

### Viewing port Uptime

**MIB:** SNMPv2-MIB, IF-MIB

**Tables used:**

sysUpTime — 1.3.6.1.2.1.1.3

ifLastChange — 1.3.6.1.2.1.2.2.1.9

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.1.3  
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.2.2.1.9.{ifindex}
```

#### **Example of viewing GigabitEthernet 1/0/2 port Uptime**

CLI command:

```
show interface status GigabitEthernet 1/0/2
```

SNMP command:



```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.1.3
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.2.2.1.9.50
```



**The output of the first command must be removed from the output of the second command. The resulting value will be the Uptime of the port.**

### Viewing the switch's Uptime counter in seconds for a stack unit



**The team is relevant for Uptime less than 136 years.**

**MIB:** ELTEX-PHYSICAL-DESCRIPTION-MIB.mib

**Tables used:** eltPhdUnitEnvParamTable — 1.3.6.1.4.1.35265.1.23.53.7

```
snmpget -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.53.7.1.1.{unit}
```

### **Example of viewing the switch's Uptime counter in seconds for a stack unit**

```
SNMP command:
snmpget -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.53.7.1.1.1
```

### Enabling CPU traffic monitoring service

**MIB:** rlstct.mib

**Tables used:** rlSctCpuRateEnabled — 1.3.6.1.4.1.89.203.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.203.1.0 i {true(1), false(2)}
```

### **Example of enabling CPU traffic monitoring service**

```
CLI command:
service cpu-input-rate
```

```
SNMP command:
snmpset -v2c -c private 192.168.1.30 1.3.6.1.4.1.89.203.1.0 i 1
```

### Viewing statistics on the speed of incoming frames processed by the CPU

**MIB:** rlstct.mib

**Tables used:** rlSctCpuRate — 1.3.6.1.4.1.89.90.1.2.1.3

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.203.2
```

### Example of viewing statistics on the speed of incoming frames processed by the CPU

```
CLI command:  
show cpu input-rate  
  
SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.203.2
```

### Viewing the counters and the number of packets processed by CPU per second (by traffic type)

**MIB:** rlsct.mib

**Tables used:** eltCpuRateStatisticsTable — 1.3.6.1.4.1.35265.1.23.1.773.1.2.1

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.1.23.1.773.1.2.1.1.{rate in pps(2), packets count(3)}
```

### Example of viewing the number of packets processed by CPU per second

```
CLI command:  
show cpu input-rate detailed  
  
SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.1.773.1.2.1.1.2
```



### Assigning indexes to traffic types:

- stack(1)
- http(2)
- telnet(3)
- ssh(4)
- snmp(5)
- ip(6)
- arp(7)
- arpInspec(8)
- stp(9)
- ieee(10)
- routeUnknown(11)
- ipHopByHop(12)
- mtuExceeded(13)
- ipv4Multicast(14)
- ipv6Multicast(15)
- dhcpSnooping(16)
- igmpSnooping(17)
- mldSnooping(18)
- ttlExceeded(19)
- ipv4IllegalAddress(20)
- ipv4HeaderError(21)
- ipDaMismatch(22)
- sflow(23)
- logDenyAces(24)
- dhcpv6Snooping(25)
- vrrp(26)
- logPermitAces(27)
- ipv6HeaderError (28)

## Changing CPU limits

**MIB:** eltSwitchRateLimiterMIB.mib

**Tables used:** eltCPURateLimiterTable — 1.3.6.1.4.1.35265.1.23.1.773.1.1.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.1.773.1.1.1.2.{index} i {limiter value}
```

### **Example of setting a 512 pps limit for snmp traffic to CPU**

CLI command:

```
service cpu-rate-limits snmp 512
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.1.773.1.1.1.2.4 i 512
```



### **Index list:**

- eltCPURLTypeHttp(1)**
- eltCPURLTypeTelnet(2)**
- eltCPURLTypeSsh(3)**
- eltCPURLTypeSnmp(4)**
- eltCPURLTypeIp(5)**
- eltCPURLTypeLinkLocal(6)**
- eltCPURLTypeArpRouter(7)**
- eltCPURLTypeArpInspection(9)**
- eltCPURLTypeStpBpdu(10)**
- eltCPURLTypeOtherBpdu(11)**
- eltCPURLTypeIpRouting(12)**
- eltCPURLTypeIpOptions(13)**
- eltCPURLTypeDhcpSnoop(14)**
- eltCPURLTypeIcmpSnoop(16)**
- eltCPURLTypeMldSnoop(17)**
- eltCPURLTypeSflow(18)**
- eltCPURLTypeLogDenyAces(19)**
- eltCPURLTypeIpErrors(20)**
- eltCPURLTypeOther(22)**

### CPU utilization monitoring

**MIB:** rlmng.mib

**Tables used:**

rlCpuUtilDuringLastSecond — 1.3.6.1.4.1.89.1.7

rlCpuUtilDuringLastMinute — 1.3.6.1.4.1.89.1.8

rlCpuUtilDuringLastMinute — 1.3.6.1.4.1.89.1.8

- Download in the last five seconds: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.1.7
- Load for the last minute: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.1.8
- Download in 5 minutes: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.1.9

**Example of viewing CPU utilization for the last 5 seconds**

```
CLI command:
show cpu utilization

SNMP command:
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.1.7
```

### Enable CPU utilization monitoring by tasks

**MIB:** RADLAN-rndMng

**Tables used:** rlCpuTasksUtilEnable — 1.3.6.1.4.1.89.1.6

```
snmpset -v2c -c <community> <IP address>
1.3.6.1.4.1.89.1.6.0 i {true(1), false(2)}
```

**Enable CPU utilization monitoring by tasks**

```
CLI command:
service tasks-utilization

SNMP command:
snmpset -v2c -c private 192.168.1.30 1.3.6.1.4.1.89.1.6.0 i 1
```

### CPU utilization monitoring by tasks

**MIB:** ELTEX-MES-MNG-MIB

**Tables used:**

eltCpuTasksUtilStatisticsUtilizationDuringLast5Seconds — 1.3.6.1.4.1.35265.1.23.1.9.1.2.1.1.3,

eltCpuTasksUtilStatisticsUtilizationDuringLastMinute — 1.3.6.1.4.1.35265.1.23.1.9.1.2.1.1.4,

eltCpuTasksUtilStatisticsUtilizationDuringLast5Minutes — 1.3.6.1.4.1.35265.1.23.1.9.1.2.1.1.5

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.1.9.1.2.1.1.3.{5sec(3), 1min(4), 5min(5)}.{task index}
```

**Example of viewing utilization by tasks for the last 5 seconds**

```
CLI command:
show tasks utilization

SNMP command:
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.1.9.1.2.1.1.3
```



**Linking indexes to tasks:**

LTMR(0)	NTST(50)	IPRD(100)
ROOT(1)	CNLD(51)	PNGA(101)
IT33(2)	HOST(52)	UDPR(102)
IV11(3)	TBI_(53)	VRRP(103)
URGN(4)	BRMN(54)	TRCE(104)
TMNG(5)	COPY(55)	SSLP(105)
IOTG( 6 )	TRNS(56)	WBSO(106)
IOUR(7)	MROR( 57)	WBSR(107)
IOTM( 8)	DFST(58)	GOAH(108)
SSHU(9)	SFTR(59)	ECHO(109)
XMOD(10)	SFMG(60)	TNSR(110)
MSCm(11)	HCPT(61)	TNSL(111)
STSA(12)	EVAU(62)	SSHP(112)
STSB(13 )	EVFB(63 )	PTPT(113)
STSC(14)	EVRT(64)	NBBT(114)
STSD(15)	EPOE(65)	SQIN(115)
STSE(16 )	DSPT(66)	MUXT(116)
CPUT(17)	B_RS(67)	DMNG(117)
EVAP(18)	TRIG(68)	DSYN(118)
HCLT(19)	MACT(69)	HSEU(119)
EVLC(20 )	SW2M(70)	DTSA(120)
SELC(21)	3SWQ(71)	SS2M(121)
SEAU(22)	POLI(72)	DSND(122)
ESTC(23)	OBSR(73)	STMB(123)
SSTC(24)	NTPL(74)	AAAT(124)
BOXS(25)	L2HU(75)	AATT(125)
BSNC(26)	L2PS(76)	SCPT(126)
BOXM(27)	SFSM(77)	DH6C(127)
TRMT(28)	NSCT(78)	RCLA(128)
D_SP(29 )	NSFP(79)	RCLB(129)
D_LM(30)	NVCT(80 )	RCDS(130)
PLCT(31)	NACT(81)	GRN_(131)
PLCR(32)	NSTM(82)	IPMT(132)
exRX(33)	NINP(83)	SNTP(133)
3SWF(34)	L2UT(84)	DHCP(134)
MSRP(35)	BRGS(85)	DHCp(135)
HSES(36)	FHSS(86)	RELY(136)
HSCS(37)	FHSF(87)	MSSS(137)
MRDP(38)	FFTT(88)	WBAM(138)
MLDP(39 )	IPAT(89)	WNTT(139)
SETX( 40)	IP6M(90)	RADS(140)
EVTX(41)	IP6L(91)	SNAS(141)
SERX(42)	IP6C(92)	SNAE(142)
EVRX(43)	IP6R(93)	SNAD(143)
HLTX(44)	RPTS(94)	MNGT(144)
LBDR(45)	ARPG(95)	UTST(145)
DDFG(46)	IPG_(96 )	SOCK(146)
SYLG(47)	DNSC(97)	TCPP(147)
CDB_(48)	ICMP(98 )	UNQt(148)
SNMP(49)	TFTP(99)	

### Viewing the total amount of RAM

**MIB:** ELTEX-PROCESS-MIB.mib

**Tables used:** eltexProcessMemoryTotal — 1.3.6.1.4.1.35265.41.1.2.1.1.3

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.41.1.2.1.1.3.0
```

#### **Example of viewing the total amount of RAM**

CLI command:

```
show cpu utilization
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.41.1.2.1.1.3.0
```

### Viewing the free amount of RAM

**MIB:** ELTEX-PROCESS-MIB.mib

**Tables used:** eltexProcessMemoryFree — 1.3.6.1.4.1.35265.41.1.2.1.1.7

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.41.1.2.1.1.7.0
```

#### **Example of viewing the free amount of RAM**

CLI command:

```
show cpu utilization
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.41.1.2.1.1.7.0
```

### Enabling jumbo-frame support

**MIB:** radlan-jumboframes-mib.mib

**Tables used:** rJumboFrames — 1.3.6.1.4.1.89.91

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.91.2.0 i {enabled(1), disabled(2)}
```

#### **Example of enabling jumbo-frame support**

CLI command:

```
port jumbo-frame
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.91.2.0 i 1
```

## 4.2 System parameters

### Power supply unit state control

**MIB:** rphysdescription.mib

**Tables used:** rIPhdUnitEnvParamTable — 1.3.6.1.4.1.89.53.15

- The primary power supply unit: `snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.53.15.1.2`
- The redundant power supply unit: `snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.53.15.1.3`

### **Example of viewing the primary power supply unit state**

CLI command:

```
show system
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.53.15.1.2
```



**The primary power supply unit has the following states:**

normal (1)  
warning (2)  
critical (3)  
shutdown (4)  
notPresent (5)  
notFunctioning (6)

**The redundant power supply unit has the following states:**

normal (1)  
warning (2)  
critical (3)  
shutdown (4)  
notPresent (5)  
notFunctioning (6)

### Control of the type of the power supply unit installed

**MIB:** rEnv.mib

**Tables used:** rEnvMonSupplySource — 1.3.6.1.4.1.89.83.1.2.1.4

```
snmpwalk -v2c -c <community> <IP address> \ 1.3.6.1.4.1.89.83.1.2.1.4
```

### **Example of requesting the type of the power supply unit installed**

CLI command:

```
show system power-supply
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.83.1.2.1.4
```



**The type can take the following values:**

- 1 — the power supply is not installed, or the type is not defined**
- 2 — AC power supply**
- 3 — DC power supply.**

**Battery status monitoring**

**MIB:** eltEnv.mib

**Tables used:** eltEnvMonBatteryState — 1.3.6.1.4.1.35265.1.23.11.1.1.2

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.1.23.11.1.1.2
```

**Example of viewing the battery status**

```
CLI command:  
show system battery  
  
SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.11.1.1.2
```



**Possible states:**

- normal (1) — battery charged**
- warning (2) — battery is discharging**
- critical (3) — battery is low**
- notPresent (5) — no battery**
- notFunctioning (6) — battery current breaker failure**
- restore (7) — battery being charged**

**Battery charge level monitoring**

**MIB:** eltEnv.mib

**Tables used:** eltEnvMonBatteryStatusCharge — 1.3.6.1.4.1.35265.1.23.11.1.1.3

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.1.23.11.1.1.3
```

**Example of viewing the battery charge level**

```
CLI command:  
show system battery  
  
SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.11.1.1.3
```



### Fans state control

**MIB:** rphysdescription.mib

**Tables used:** rIPhdUnitEnvParamTable — 1.3.6.1.4.1.89.53.15

- Fan 1: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.53.15.1.4
- Fan 2: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.53.15.1.5
- Fan 3: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.53.15.1.6
- Fan 4: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.53.15.1.7

#### **Example of viewing the state of MES3324F fan 3**

CLI command:  
show system

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.53.15.1.6



**The following states are possible:**

- normal (1)**
- notFunctioning (5)**

### Monitoring of temperature sensor readings

**MIB:** rphysdescription.mib

**Tables used:** rIPhdUnitEnvParamTable — 1.3.6.1.4.1.89.53.15

- Temperature sensor 1: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.53.15.1.10

#### **Example of viewing the sensor temperature**

CLI command:  
show system

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.53.15.1.10



**MES5324 has 4 temperature sensors, the readings of which can be viewed by the CLI command: show system sensors.**

**SNMP command:**

**Sensor 1:**  
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.83.2.1.1.1.4.68420481

**Sensor 2:**  
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.83.2.1.1.1.4.68420482

**Sensor 3:**  
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.83.2.1.1.1.4.68420483

**Sensor 4:**  
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.83.2.1.1.1.4.68420484

### Monitoring of temperature sensors state:

**MIB:** rphysdescription.mib

**Tables used:** rPhdUnitEnvParamTable — 1.3.6.1.4.1.89.53.15

Temperature sensor 1: snmpwalk -v2c -c <community> <IP address> 1.3.6.1.4.1.89.53.15.1.11

#### **Example of viewing the state of temperature sensors**

CLI command:  
show system sensors

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.53.15.1.11



**MES5324 has 4 temperature sensors, the readings of which can be viewed by the CLI command: show system sensors.**

#### SNMP command:

**Sensor 1:**

snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.83.2.1.1.1.5.68420481

**Sensor 2:**

snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.83.2.1.1.1.5.68420482

**Sensor 3:**

snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.83.2.1.1.1.5.68420483

**Sensor 4:**

snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.83.2.1.1.1.5.68420484



#### Temperature sensors states:

ok (1)

unavailable (2)

nonoperational (3)

## **4.3 Stack parameters**

### Stack parameters monitoring

**MIB:** rphysdescription.mib

**Tables used:** rPhdStackTable — 1.3.6.1.4.1.89.53.4

snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.53.4

### Example of viewing stack parameters

CLI command:

```
show stack
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.53.4
```

### Stack ports monitoring

**MIB:** rphysdescription.mib

**Tables used:** rICascadeTable — 1.3.6.1.4.1.89.53.23

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.53.23
```

### Example of viewing stack ports status

CLI command:

```
show stack links
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.53.23
```

## 4.4 Device management

### Setting/changing the hostname on the device

**MIB:** SNMPv2-MIB

**Tables used:** sysName — 1.3.6.1.2.1.1.5

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.2.1.1.5.0 s "{hostname}"
```

### Example of "mes2324" hostname assignment

CLI command:

```
hostname mes2324
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.2.1.1.5.0 s "mes2324"
```

### Enabling/disabling management acl

**MIB:** RADLAN-MNGINF-MIB

**Tables used:**

rIMngInfEnable — 1.3.6.1.4.1.89.89.2

rIMngInfActiveListName — 1.3.6.1.4.1.89.89.3

```
snmpset -v2c -c <community> <IP address>
1.3.6.1.4.1.89.89.2.0 i {true(1), false(2)}\
1.3.6.1.4.1.89.89.3.0 s {name}
```

### Example of "eltex" management acl enabling

CLI command:

```
management access-class eltex
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.89.2.0 i 1 \
1.3.6.1.4.1.89.89.3.0 s eltex
```

### Using the ping utility

**MIB:** rlapplcation.mib

**Tables used:** rsPingInetTable — 1.3.6.1.4.1.89.35.4.2

```
snmpset -v2c -c <community> <IP address>\
1.3.6.1.4.1.89.35.4.1.1.2.{IP address}> i {Packet count}\
1.3.6.1.4.1.89.35.4.1.1.3.{IP address}> i {Packet Size}\
1.3.6.1.4.1.89.35.4.1.1.4.{IP address}> i {Packet Timeout}\
1.3.6.1.4.1.89.35.4.1.1.5.{IP address}> i {Ping Delay}\
1.3.6.1.4.1.89.35.4.1.1.6.{IP address}> i {Send SNMP Trap(2)}\
1.3.6.1.4.1.89.35.4.1.1.14.{IP address}> i {createAndGo(4), destroy(6),
active(1)}
```

### Example of a 192.168.1.1 node ping

CLI command:

```
ping 192.168.1.1 count 10 size 250 timeout 1000
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.35.4.1.1.2.192.168.1.1 i 10 \
1.3.6.1.4.1.89.35.4.1.1.3.192.168.1.1 i 250 \
1.3.6.1.4.1.89.35.4.1.1.4.192.168.1.1 i 1000 \
1.3.6.1.4.1.89.35.4.1.1.5.192.168.1.1 i 0 \
1.3.6.1.4.1.89.35.4.1.1.6.192.168.1.1 i 2 \
1.3.6.1.4.1.89.35.4.1.1.14.192.168.1.1 i 4
```



**When 4 (createAndGo) is set to the rsPingEntryStatus field, a ping operation is created and enabled.**

**To re-ping a remoted host, set the value 1 (active) in the rsPingEntryStatus field.**

**After the operation is completed, delete all the entries by setting the value 6 (destroy) in the rsPingEntryStatus field. Otherwise, it will be impossible to ping another host via CLI and SNMP.**

**Example of removal:**

```
snmpset -v2c -c private 192.168.1.30\
1.3.6.1.4.1.89.35.4.1.1.2.192.168.1.1 i 10\
1.3.6.1.4.1.89.35.4.1.1.3.192.168.1.1 i 250\
1.3.6.1.4.1.89.35.4.1.1.4.192.168.1.1 i 1000\
1.3.6.1.4.1.89.35.4.1.1.5.192.168.1.1 i 0\
1.3.6.1.4.1.89.35.4.1.1.6.192.168.1.1 i 2\
1.3.6.1.4.1.89.35.4.1.1.14.192.168.1.1 i 6
```

## Ping utility monitoring

**MIB:** rapplication.mib

**Tables used:** rsPingEntry — 1.3.6.1.4.1.89.35.4.1.1

```
snmpwalk -v2c -c <community> <IP address>\
```

```
1.3.6.1.4.1.89.35.4.1.1.{Number of packets sent(7), Number of packets
received(8), Minimum response time(9), Average response time(10), Maximum
response time(11)}
```

### Example of viewing the number of packets received

CLI command:

```
ping 192.168.1.31
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.35.4.1.1.8
```



**When the value 6 (destroy) is set in the rsPingEntryStatus field, monitoring will be forbidden until a new operation is created.**

## System log configuration

**MIB:** DRAFT-IETF-SYSLOG-DEVICE-MIB

**Tables used:** snmpSyslogCollectorEntry — 1.3.6.1.4.1.89.82.1.2.4.1

```
snmpset -v2c -c <community> -t 10 -r 5 <IP address> \
1.3.6.1.4.1.89.82.1.2.4.1.2.1 s "{name}" \
1.3.6.1.4.1.89.82.1.2.4.1.3.1 i {ipv4(1), ipv6(2)} \
1.3.6.1.4.1.89.82.1.2.4.1.4.1 x {ip add in HEX} \
1.3.6.1.4.1.89.82.1.2.4.1.5.1 u {udp port number} \
1.3.6.1.4.1.89.82.1.2.4.1.6.1 i {syslog facility(16-24)} \
1.3.6.1.4.1.89.82.1.2.4.1.7.1 i {severity level} \
1.3.6.1.4.1.89.82.1.2.4.1.9.1 i {createAndGo(4), destroy(6)}
```

### Example of adding a server for logging

CLI command:

```
logging host 192.168.1.1 description 11111
```

SNMP command:

```
snmpset -v2c -c private -t 10 -r 5 192.168.1.30 \
1.3.6.1.4.1.89.82.1.2.4.1.2.1 s "11111" \
1.3.6.1.4.1.89.82.1.2.4.1.3.1 i 1 \
1.3.6.1.4.1.89.82.1.2.4.1.4.1 x C0A80101 \
1.3.6.1.4.1.89.82.1.2.4.1.5.1 u 514 \
1.3.6.1.4.1.89.82.1.2.4.1.6.1 i 23 \
1.3.6.1.4.1.89.82.1.2.4.1.7.1 i 6 \
1.3.6.1.4.1.89.82.1.2.4.1.9.1 i 4
```



Severity level is specified as follows:

emergency(0),  
alert(1),  
critical(2),  
error(3),  
warning(4),  
notice(5),  
info(6),  
debug(7)

Facility:

local0(16),  
local1(17),  
local2(18),  
local3(19),  
local4(20),  
local5(21),  
local6(22),  
local7(23),  
no-map(24)

## 5 SYSTEM TIME CONFIGURATION

### SNTP server address configuration

**MIB:** rlsntp.mib

**Tables used:** rlSntpConfigServerInetTable — 1.3.6.1.4.1.89.92.2.2.17

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.92.2.2.17.1.3.1.4.{ip address in DEC. IP address bytes are
  separated by points} i {true(1), false(2). Set poll value} \
  1.3.6.1.4.1.89.92.2.2.17.1.9.1.4.{ip address in DEC. IP address bytes are
  separated by points} u 0 \
  1.3.6.1.4.1.89.92.2.2.17.1.10.1.4.{ip address in DEC. IP address bytes are
  separated by points} i {createAndGo(4), destroy(6)}
```

### **Example of specifying SNTP server with IP address 91.226.136.136**

**CLI command:**  
sntp server 91.226.136.136 poll

**SNMP command:**  
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.92.2.2.17.1.3.1.4.91.226.136.136 i 1 \  
1.3.6.1.4.1.89.92.2.2.17.1.9.1.4.91.226.136.136 u 0 \  
1.3.6.1.4.1.89.92.2.2.17.1.10.1.4.91.226.136.136 i 4

### Setting the polling time for SNTP client

**MIB:** rlsntp.mib

**Tables used:** rlSntpNtpConfig — 1.3.6.1.4.1.89.92.2.1

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.92.2.1.4.0 i {range 60-86400}
```

### **Example of setting the polling time of 60 seconds**

**CLI command:**  
sntp client poll timer 60

**SNMP command:**  
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.92.2.1.4.0 i 60



**To return to default settings, set the time of 1024 seconds.**

### Setting the operation of unicast SNTP clients

**MIB:** rlsntp.mib

**Tables used:** rlSntpConfig — 1.3.6.1.4.1.89.92.2.2

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.92.2.2.5.0 i {true(1), false(2)}
```

### Example of a unicast SNMP server polling

CLI command:

```
sntp unicast client poll
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.92.2.2.5.0 i 1
```

### Adding a time zone

**MIB:** rlsntp.mib

**Tables used:** rlTimeSyncMethodMode — 1.3.6.1.4.1.89.92.1

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.92.1.6.0 s "{TimeZone}" \  
1.3.6.1.4.1.89.92.1.7.0 s "{NameZone}"
```

### Example of adding a time zone on a device

CLI command:

```
clock timezone test +7
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.92.1.6.0 s "+7:00" \  
1.3.6.1.4.1.89.92.1.7.0 s "test"
```



## 6 INTERFACE CONFIGURATION

### 6.1 Ethernet interface parameters

#### Viewing Description port

**MIB:** IF-MIB or eltMng.mib

**Tables used:** ifAlias — 1.3.6.1.2.1.31.1.1.1.18 or iflongDescr — 1.3.6.1.4.1.35265.1.23.1.1.31.1.1.1.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.31.1.1.1.18.{ifIndex}
```

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.1.1.31.1.1.1.1.{ifIndex}
```

#### **Example of viewing Description on GigabitEthernet 1/0/1 interface**

CLI command:

```
show interfaces description GigabitEthernet 1/0/1
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.31.1.1.1.18.49
```

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.1.1.31.1.1.1.1.49
```

#### Viewing Description vlan

**MIB:** Q-BRIDGE-MIB

**Tables used:** dot1qVlanStaticTable — 1.3.6.1.2.1.17.7.1.4.3

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.17.7.1.4.3.1.1.{vlan id}
```

#### **Example of viewing Description vlan 100**

CLI command:

```
show interfaces description vlan 100
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.17.7.1.4.3.1.1.100
```

#### Viewing speed on the interface

**MIB:** IF-MIB

**Tables used:** ifHighSpeed — 1.3.6.1.2.1.31.1.1.1.15

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.31.1.1.1.15.{ifIndex}
```

### Example of a speed viewing on GigabitEthernet1/0/2

```
CLI command:
show interface status GigabitEthernet 1/0/2
SNMP command:
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.31.1.1.1.15.50
```

### Enabling/disabling speed autonegotiation on an interface

**MIB:** rlinterfaces.mib

**Tables used:** swIfSpeedDuplexAutoNegotiation — 1.3.6.1.4.1.89.43.1.1.16

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.43.1.1.16.{ifIndex} i {negotiation(1), no negotiation(2)}
```

### Example of enabling negotiation on GigabitEthernet1/0/2

```
CLI command:
interface GigabitEthernet 1/0/2
no negotiation

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.43.1.1.16.50 i 2
```

### Enabling an autonegotiation procedure omitting, if a partner on the opposite side does not respond

**MIB:** eltinterfaces.mib

**Tables used:** eltSwifAutoNegotiationBypass — 1.3.6.1.4.1.35265.1.23.43.1.1.3

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.43.1.1.3.{ifIndex} i {negotiationbypass(1), no
negotiation bypass(2)}
```

### Example of enabling negotiation on TenGigabitEthernet1/0/2

```
CLI command:
interface TenGigabitEthernet 1/0/2
no negotiation bypass

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.43.1.1.3.106 i 2
```

### Setting speed autonegotiation modes on an interface

**MIB:** rlinterfaces.mib

**Tables used:** swIfAdminSpeedDuplexAutoNegotiationLocalCapabilities — 1.3.6.1.4.1.89.43.1.1.40

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.43.1.1.40.{ifIndex} x {negotiation mode(HEX)}
```

### Example of 10f and 100f autonegotiation on GigabitEthernet1/0/2

CLI command:

```
interface GigabitEthernet 1/0/2
negotiation 10f 100f
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.43.1.1.40.50 x 14
```



In the binary system, 10f and 100f are written as 00010100. In the hexadecimal system they are written as 14.

#### Bit description

default(0),  
unknown(1),  
tenHalf(2),  
tenFull(3),  
fastHalf(4),  
fastFull(5),  
gigaHalf(6),  
gigaFull(7)

#### Bit order

0 1 2 3 4 5 6 7

#### Viewing a port duplex mode

MIB: EtherLike-MIB

Tables used: dot3StatsDuplexStatus — 1.3.6.1.2.1.10.7.2.1.19

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.10.7.2.1.19.{ifindex}
```

#### Example of viewing the GigabitEthernet 1/0/1 duplex mode

CLI command:

```
show interfaces status GigabitEthernet 1/0/1
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.10.7.2.1.19.49
```



#### Description of the output values

unknown (1)  
halfDuplex (2)  
fullDuplex (3)

### Changing a duplex mode on the interface

**MIB:** RADLAN-rlInterfaces

**Tables used:** swIfDuplexAdminMode — 1.3.6.1.4.1.89.43.1.1.3

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.43.1.1.3.{ifIndex} i {none(1),half(2),full (3)}
```

#### **Example of changing the GigabitEthernet 1/0/1 duplex mode**

CLI command:

```
interface GigabitEthernet 1/0/1  
duplex half
```

SNMP command:

```
snmpset-v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.43.1.1.3.49 i 2
```

### Viewing an interface transmission medium

**MIB:** EtherLike-MIB

**Tables used:** swIfTransceiverType — 1.3.6.1.4.1.89.43.1.1.7

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.43.1.1.7.{ifindex}
```

#### **Example of viewing the GigabitEthernet 1/0/1 transmission medium**

CLI command:

```
show interfaces status GigabitEthernet 1/0/1
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.43.1.1.7.49
```



#### Description of the output values

**Copper (1)**

**fiberOptics (2)**

**ComboCopper (3)**

**comboFiberOptics (4)**

### Flow control

**MIB:** RADLAN-rlInterfaces

**Tables used:** swIfFlowControlMode — 1.3.6.1.4.1.89.43.1.1.14

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.43.1.1.14.{ifindex} i {on(1),off(2),auto (3)}
```

### Example of enabling flow control on the GigabitEthernet1/0/2 interface

```

CLI command:
interface GigabitEthernet 1/0/2
flowcontrol on

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.43.1.1.14.50 i 1

```

### Viewing an administrative state of the port

**MIB:** IF-MIB

**Tables used:** ifAdminStatus — 1.3.6.1.2.1.2.2.1.7

```

snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.2.2.1.7.{ifIndex}

```

### Example of viewing the GigabitEthernet 1/0/1 port state

```

CLI command:
show interfaces status GigabitEthernet 1/0/1

SNMP command:
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.2.2.1.7.49

```



#### Possible options:

- up(1)**
- down(2)**
- testing(3)**

### Enabling/disabling a configured interface

**MIB:** IF-MIB

**Tables used:** ifAdminStatus — 1.3.6.1.2.1.2.2.1.7

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.2.1.2.2.1.7.{ifIndex} i {up(1),down(2)}

```

### Example of disabling the GigabitEthernet interface 1/0/1

```

CLI command:
interface GigabitEthernet 1/0/1
shutdown

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.2.1.2.2.1.7.49 i 2

```

### Viewing the operative state of the port

**MIB:** IF-MIB

**Tables used:** ifOperStatus — 1.3.6.1.2.1.2.2.1.8

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.2.2.1.8.{ifIndex}
```

#### **Example of viewing the GigabitEthernet 1/0/1 port state**

CLI command:

```
show interfaces status GigabitEthernet 1/0/1
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.2.2.1.8.49
```



#### **Possible options:**

**up(1)**

**down(2)**

### Determining a port connection type

**MIB:** rlinterfaces.mib

**Tables used:** swIfTransceiverType — 1.3.6.1.4.1.89.43.1.1.7

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.43.1.1.7.{ifindex}
```

#### **Example of determining the GigabitEthernet 1/0/1 port connection type**

CLI command:

```
show interfaces status
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.43.1.1.7.49
```



#### **Possible options:**

**regular (1)**

**fiberOptics (2)**

**comboRegular (3)**

**comboFiberOptics (4)**

### Viewing the counter of unicast packets on the interface

**MIB:** IF-MIB

**Tables used:** ifInUcastPkts — 1.3.6.1.2.1.2.2.1.11

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.2.2.1.11.{ifIndex}
```

### Example of viewing an incoming unicast packets counter on the GigabitEthernet 1/0/2 interface

CLI command:

```
show interface counters GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.2.2.1.11.50
```

### Viewing the counter of multicast packets on the interface

**MIB:** IF-MIB

**Tables used:** ifInMulticastPkts — 1.3.6.1.2.1.31.1.1.1.2

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.31.1.1.1.2.{ifindex}
```

### Example of viewing an incoming multicast packets counter on the GigabitEthernet 1/0/2 interface

CLI command:

```
show interface counters GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.31.1.1.1.2.50
```

### Viewing the counter of broadcast packets on the interface

**MIB:** IF-MIB

**Tables used:** ifInBroadcastPkts — 1.3.6.1.2.1.31.1.1.1.3

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.31.1.1.1.3.{ifindex}
```

### Example of viewing an incoming broadcast packets counter on the GigabitEthernet 1/0/2 interface

CLI command:

```
show interface counters GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.31.1.1.1.3.50
```

### Viewing the octet counter on the interface

**MIB:** IF-MIB

**Tables used:**

ifInOctets — 1.3.6.1.2.1.2.2.1.10

ifHCInOctets - 1.3.6.1.2.1.31.1.1.1.6

ifOutOctets— 1.3.6.1.2.1.2.2.1.16

ifHCOctets - 1.3.6.1.2.1.31.1.1.1.10

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.2.2.1.10.{ifindex}
```

### Example of viewing the received octets counter on GigabitEthernet 1/0/2 interface

CLI command:

```
show interface counters gigabitethernet1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.2.2.1.10.50
```



**An octet is the number of bytes.**

**1 octet = 1 byte**

### Viewing FCS Errors counter on an interface

**MIB:** EtherLike-MIB

**Tables used:** dot3StatsFCSErrors — 1.3.6.1.2.1.10.7.2.1.3

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.10.7.2.1.3.{ifindex}
```

### Example of viewing the FCS Errors counter on the GigabitEthernet 1/0/2 interface

CLI command:

```
show interface counters gigabitethernet1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.10.7.2.1.3.50
```

### Viewing the Internal MAC Rx Errors counter on an interface

**MIB:** EtherLike-MIB

**Tables used:** dot3StatsInternalMacReceiveErrors — 1.3.6.1.2.1.10.7.2.1.16

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.10.7.2.1.16.{ifindex}
```

### Example of viewing the MAC Rx Errors counter on the GigabitEthernet 1/0/2 interface

CLI command:

```
show interface counters GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.10.7.2.1.16.50
```

### Example of viewing the Transmitted Pause Frames counter on an interface

**MIB:** EtherLike-MIB

**Tables used:** dot3OutPauseFrames — 1.3.6.1.2.1.10.7.10.1.4

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.10.7.10.1.4.{ifindex}
```







1) A bit mask is set to the stack counters reset value for all ports of all units of the stack:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.54.4.0 x
000000000000FFFFFF00000000F00000000000FFFFFF00000000F
000000000000FFFFFF00000000F00000000000FFFFFF00000000F000000000
00FFFFFF0
000000F0000000000000FFFFFF00000000F00000000000FFFFFF00000000F00
00000000
0FFFFFF00000000F000000000000000000000000000000000000000000000000001FFFE0000000000
```

2) To view the value of a bit mask, use the following command:

```
snmpwalk -v2c -c public <IP address> \
1.3.6.1.4.1.89.54.9.0
```

### Monitoring of switch ports load

**MIB:** eltMes.mib

**Tables used:** eltSwIfUtilizationEntry — 1.3.6.1.4.1.35265.1.23.43.2.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.43.2.1.{parameter}
```

### **Example of viewing port load**

```
CLI command:
show interfaces utilization

SNMP command:
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.43.2.1.1
```



### The list of possible parameters:

```
eltSwIfUtilizationIfIndex(1)
eltSwIfUtilizationAverageTime(2)
eltSwIfUtilizationCurrentInPkts(3)
eltSwIfUtilizationCurrentInRate(4)
eltSwIfUtilizationCurrentOutPkts(5)
eltSwIfUtilizationCurrentOutRate(6)
eltSwIfUtilizationAverageInPkts(7)
eltSwIfUtilizationAverageInRate(8)
eltSwIfUtilizationAverageOutPkts(9)
eltSwIfUtilizationAverageOutRate(10)
```





### Ban default vlan on a port

**MIB:** eltVlan.mib

**Tables used:** eltVlanDefaultForbiddenPorts — 1.3.6.1.4.1.35265.1.23.5.5.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.5.5.1.0 x {port as a bit mask}
```

#### **Example of banning the default vlan on the GigabitEthernet1/0/5 port**

CLI command:

```
interface GigabitEthernet 1/0/5
switchport forbidden default-vlan
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.5.5.1.0 x 0000000000000008
```



**1. An example of creating a bit mask is given in section «APPENDIX A. Bit mask calculation method».**

**2. A bit mask should contain at least 10 characters.**

### Viewing VLAN name

**MIB:** rlvlan.mib

**Tables used:** rldot1qVlanStaticName — 1.3.6.1.4.1.89.48.70.1.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.70.1.1.{vlan}
```

#### **Example of viewing vlan 5 name**

CLI command:

```
show vlan tag 5
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.48.70.1.1.5
```

### Viewing port membership in VLAN

**MIB:** rlvlan.mib

**Tables used:** rldot1qPortVlanStaticTable — 1.3.6.1.4.1.89.48.68

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.68.1.{1-4}.{ifindex}
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.68.1.{5-8}.{ifindex}
```

### Example of vlan viewing on GigabitEthernet 1/0/5

CLI command:

```
show interfaces switchport GigabitEthernet 1/0/5
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.48.68.1.1.54
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.48.68.1.5.54
```



1. The example shows 2 snmpwalk commands. If a port is Tagged, values in the second command output are zero, and a vlan number corresponds to the first command output values. If a port is Untagged, the second command output contains values other than zero, and a vlan number corresponds to these values.

2. Table list:

```
rldot1qPortVlanStaticEgressList1to1024 — 1.3.6.1.4.1.89.48.68.1.1.{ifindex}
rldot1qPortVlanStaticEgressList1025to2048 — 1.3.6.1.4.1.89.48.68.1.2.{ifindex}
rldot1qPortVlanStaticEgressList2049to3072 — 1.3.6.1.4.1.89.48.68.1.3.{ifindex}
rldot1qPortVlanStaticEgressList3073to4094 — 1.3.6.1.4.1.89.48.68.1.4.{ifindex}
rldot1qPortVlanStaticUntaggedEgressList1to1024 — 1.3.6.1.4.1.89.48.68.1.5.{ifindex}
rldot1qPortVlanStaticUntaggedEgressList1025to2048
— 1.3.6.1.4.1.89.48.68.1.6.{ifindex}
rldot1qPortVlanStaticUntaggedEgressList2049to3072
— 1.3.6.1.4.1.89.48.68.1.7.{ifindex}
rldot1qPortVlanStaticUntaggedEgressList3073to4094
— 1.3.6.1.4.1.89.48.68.1.8.{ifindex}
```

3. The values obtained as a result of the query are a bit mask, the method of calculation of which is given in the section «APPENDIX A. Bit mask calculation method».

### Port mode configuration

**MIB:** rlvlan.mib

**Tables used:** vlanPortModeEntry — 1.3.6.1.4.1.89.48.22.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.22.1.1.{ifIndex} i {general(1), access(2), trunk(3),
customer(7)}
```

### Example of GigabitEthernet 1/0/2 interface configuration to the trunk mode

CLI command:

```
interface GigabitEthernet 1/0/2
switchport mode trunk
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.48.22.1.1.50 i 3
```

### Viewing a port mode

**MIB:** rvlan.mib

**Tables used:** vlanPortModeState — 1.3.6.1.4.1.89.48.22.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.22.1.1.{ifindex}
```

#### **Example of a mode viewing on GigabitEthernet 1/0/2**

CLI command:

```
show interfaces switchport GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.48.22.1.1.50
```



#### Possible options:

**general(1)**  
**access(2)**  
**trunk (3)**  
**customer (7)**

### Assigning pvid to an interface

**MIB:** Q-BRIDGE-MIB.mib

**Tables used:** dot1qPortVlanTable — 1.3.6.1.2.1.17.7.1.4.5

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.2.1.17.7.1.4.5.1.1.{ifindex} u {1-4094}
```

#### **Example of pvid 15 assignment for GigabitEthernet 1/0/2**

CLI command:

```
interface GigabitEthernet 1/0/2
 switchport general pvid 15
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.2.1.17.7.1.4.5.1.1.50 u 15
```

### Configuring a map mac

**MIB:** rvlan.mib

**Tables used:** vlanMacBaseVlanGroupTable — 1.3.6.1.4.1.89.48.45

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.45.1.3.{MAC address in DEC}.{mask} i {map-group number} \
1.3.6.1.4.1.89.48.45.1.4.{MAC address in DEC}.{mask} i {createAndGo(4),
destroy(6)}
```

### Example of configuring a map mac

```
CLI command:
vlan database
map mac a8:f9:4b:33:29:c0 32 macs-group 1
```

```
SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.48.45.1.3.168.249.75.51.41.192.32 i 1 \
1.3.6.1.4.1.89.48.45.1.4.168.249.75.51.41.192.32 i 4
```

### Setting a MAC-address-binding-based VLAN classification rule for an interface

**MIB:** rlvlan.mib

**Tables used:** vlanMacBaseVlanPortTable — 1.3.6.1.4.1.89.48.46.1.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.46.1.2.58.1 u {vlan} 1.3.6.1.4.1.89.48.46.1.3.58.1 i
{createAndGo(4), destroy(6)}
```

### Example of enabling a VLAN classification rule for the GigabitEthernet 1/0/10 interface

```
CLI command:
interface GigabitEthernet 1/0/10
switchport general map macs-group 1 vlan 20
```

```
SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.48.46.1.2.58.1 u 20 \
1.3.6.1.4.1.89.48.46.1.3.58.1 i 4
```

## 6.3 Errdisable state configuration and monitoring

### Viewing settings for automatic interface activation

**MIB:** rlinterfaces\_recovery.mib

**Tables used:** rErrdisableRecoveryEnable — 1.3.6.1.4.1.89.128.2.1.2

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.128.2.1.2
```

### Example of viewing settings for automatic interface activation

```
CLI command:
show errdisable recovery
```

```
SNMP command:
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.128.2.1.2
```



### Viewing the reason of port blocking

**MIB:** rErrdisableRecoveryIfReason

**Tables used:** rErrdisableRecoveryIfReason — 1.3.6.1.4.1.89.128.3.1.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.128.3.1.1
```

#### **Example of viewing the reason of port blocking**

CLI command:

```
show errdisable interfaces
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.128.3.1.1
```



#### **Possible options:**

- loopback-detection (1)**
- port-security (2)**
- dot1x-src-address (3)**
- acl-deny (4)**
- stp-bpdu-guard (5)**
- stp-loopback-guard (6)**
- unidirectional-link (7)**
- dhcp-rate-limit (8)**
- l2pt-guard (9)**
- storm-control (10)**

### Configuring automatic interface activation

**MIB:** rlinterfaces\_recovery.mib

**Tables used:** rErrdisableRecoveryEnable — 1.3.6.1.4.1.89.128.2.1.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.128.2.1.2. {index of reason} i {true(1), false(2)}
```

#### **Example of enabling automatic interface activation in case of loopback detection**

CLI command:

```
errdisable recovery cause loopback-detection
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.128.2.1.2.1 i 1
```



**Possible index of reason values, depending on the configuration type:**

- loopback detection — (1)**
- port-security — (2)**
- dot1x-src-address — (3)**
- acl-deny — (4)**
- stp-bpdu-guard — (5)**
- stp-loopback-guard (6)**
- unidirectional-link — (8)**
- storm-control — (9)**
- l2pt-guard — (11)**

### Configuring an interval for exit from the errdisable state

**MIB:** rlinterfaces\_recovery.mib

**Tables used:** rErrdisableRecoveryInterval — 1.3.6.1.4.1.89.128.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.128.1.0 i {interval 30-86400}
```

#### **Example of configuring a 30 second interval for exit from the errdisable state**

CLI command:

```
errdisable recovery interval 30
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.128.1.0 i 30
```

## **6.4 Configuring voice vlan**

### Adding voice vlan

**MIB:** RADLAN-vlanVoice-MIB

**Tables used:** vlanVoiceAdminVid — 1.3.6.1.4.1.89.48.54.8

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.54.8.0 i {vlan id}
```

#### **Example of adding voice vlan id 10**

CLI command:

```
voice vlan id 10
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.48.54.8.0 i 10
```

### Enabling voice vlan on an interface

**MIB:** RADLAN-vlanVoice-MIB

**Tables used:** vlanVoiceOUIBasedPortTable — 1.3.6.1.4.1.89.48.54.12.5

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.54.12.5.1.1.{ifIndex} i 1 \
1.3.6.1.4.1.89.48.54.12.5.1.2.{ifIndex} u {voice vlan id}
```

#### **Example of voice vlan enabling on the GigabitEthernet1/0/3 interface**

**CLI command:**  

```
interface GigabitEthernet 1/0/3
voice vlan enable
```

**SNMP command:**  

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.48.54.12.5.1.1.51 i 1 \
1.3.6.1.4.1.89.48.54.12.5.1.2.51 u 10
```

### Editing the OUI table

**MIB:** rlvlanVoice.mib

**Tables used:** vlanVoiceOUIBasedTable — 1.3.6.1.4.1.89.48.54.12.4

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.54.12.4.1.3.{OUI in DEC. Bytes are separated by points} i
{createAndGo(4), destroy(6)}
```

#### **Example of editing the OUI table**

**CLI command:**  

```
voice vlan oui-table add 002618
```

**SNMP command:**  

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.48.54.12.4.1.3.0.38.24 i 4
```

## **6.5 Configuring LLDP**

### Global LLDP enabling/disabling

**MIB:** rllldp.mib

**Tables used:** rllldpEnabled — 1.3.6.1.4.1.89.110.1.1.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.110.1.1.1.0 i {true (1), false (2)}
```

#### **Example of LLDP disabling**

**CLI command:**  

```
no lldp run
```

**SNMP command:**  

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.110.1.1.1.0 i 2
```

### Configuring lldp-med policy with the voice vlan number for tagged voice vlan traffic

**MIB:** rllldb.mib

**Tables used:** rLLdpXMedLocMediaPolicyContainerTable — 1.3.6.1.4.1.89.110.1.2.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.110.1.2.1.1.2.1 i {voice(1), voice-signaling(2), guest-voice(3),
guest-voice-signaling(4), softphone-voice(5), video-conferencing(6), streaming-
video(7), video-signaling(8)} \
1.3.6.1.4.1.89.110.1.2.1.1.3.1 i {vlan} \
1.3.6.1.4.1.89.110.1.2.1.1.4.1 i {priority} \
1.3.6.1.4.1.89.110.1.2.1.1.7.1 {true(1), false(2)} \
1 1.3.6.1.4.1.89.110.1.2.1.1.9.1 i {createAndGo(4), destroy(6)}
```

#### **Example of configuring lldp-med policy with VLAN 10, priority 4**

CLI command:

```
lldp med network-policy 1 voice vlan 10 vlan-type tagged up 4
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.110.1.2.1.1.2.1 i 1 \
1.3.6.1.4.1.89.110.1.2.1.1.3.1 i 10 \
1.3.6.1.4.1.89.110.1.2.1.1.4.1 i 4 \
1.3.6.1.4.1.89.110.1.2.1.1.7.1 i 1 \
1.3.6.1.4.1.89.110.1.2.1.1.9.1 i 4
```

### Configuring lldp-med policy for voice vlan tagged traffic

**MIB:** rllldb.mib

**Tables used:** rLLdpXMedNetPolVoiceUpdateMode — 1.3.6.1.4.1.89.110.1.7

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.110.1.7.0 i {manual(0), auto(1)}
```

#### **Example of configuring lldp-med policy in auto mode**

CLI command:

```
no lldp med network-policy voice auto
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.110.1.7.0 i 0
```

## **6.6 Configuring Flex-link**

### Enabling/disabling Flex-link on the interface

**MIB:** eltFlexLinksV2.mib

**Tables used:** eltexFlIfConfigBackUp — 1.3.6.1.4.1.35265.31.1.1.1.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.31.1.1.1.1.2.{ifIndex main port} i {ifIndex backup port}
```

### Example of enabling flex-link on the GigabitEthernet1/0/3 interface and assigning the role of backup interface to the GigabitEthernet1/0/4 interface in a flex-link pair

CLI command:

```
interface GigabitEthernet1/0/3 flex-link backup gigabitethernet1/0/4
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.31.1.1.1.2.51 i 52
```

### Example of disabling flex-link on the GigabitEthernet1/0/3 interface

CLI command:

```
interface GigabitEthernet1/0/3 no flex-link backup
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.31.1.1.1.2.51 i 0
```

### Setting up preemption mode on the interface participating in flex-link

**MIB:** eltFlexLinksV2.mib

**Tables used:** eltexFlIfConfigPreemptionMode — 1.3.6.1.4.1.35265.31.1.1.3.1.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.31.1.1.3.1.1.{ifIndex} i {off (1), forced (2), bandwidth (3)}
```

### Example of setting up preemption mode bandwidth on the GigabitEthernet1/0/1 interface

CLI command:

```
interface GigabitEthernet1/0/1 flex-link preemption mode bandwidth
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.31.1.1.3.1.1.49 i 3
```

### Setting the preemption delay time period for an interface with the flex-link function configured

**MIB:** eltFlexLinksV2.mib

**Tables used:** eltexFlIfConfigPreemptionDelay — 1.3.6.1.4.1.35265.31.1.1.3.1.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.31.1.1.3.1.2.{ifIndex} u {range 1-300}
```

### Example of setting the preemption delay time period of 50 seconds for the GigabitEthernet1/0/1 interface

CLI command:

```
interface GigabitEthernet1/0/1 flex-link preemption delay 50
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.31.1.1.3.1.2.49 u 50
```

## 7 CONFIGURING IPV4 ADDRESSING

### Creating an IP address on the interface vlan

**MIB:** rlip.mib

**Tables used:** rslpAddrEntry — 1.3.6.1.4.1.89.26.1.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.26.1.1.2.{ip address(DEC)} i {ifIndex} \
  1.3.6.1.4.1.89.26.1.1.3.{ip address(DEC)} a {netmask}
```

### **Example of configuring the address 192.168.10.30/24 on vlan 30**

**CLI command:**

```
interface vlan 30
ip address 192.168.10.30 /24
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.26.1.1.2.192.168.10.30 i 100029 \
  1.3.6.1.4.1.89.26.1.1.3.192.168.10.30 a 255.255.255.0
```

### Deleting an IP address from the interface vlan

**MIB:** rlip.mib

**Tables used:** rslpAddrEntry — 1.3.6.1.4.1.89.26.1.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.26.1.1.2.{ip address(DEC)} i {ifIndex} \
  1.3.6.1.4.1.89.26.1.1.3.{ip address(DEC)} a {netmask} \
  1.3.6.1.4.1.89.26.1.1.6.{ip address(DEC)} i 2
```

### **Example of deleting the IP address 192.168.10.30 from vlan 30**

**CLI command:**

```
interface vlan 30
no ip address 192.168.10.30
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.26.1.1.2.192.168.10.30 i 100029 \
  1.3.6.1.4.1.89.26.1.1.3.192.168.10.30 a 255.255.255.0 \
  1.3.6.1.4.1.89.26.1.1.6.192.168.10.30 i 2
```

### Obtaining an IP address via DHCP on the interface vlan

**MIB:** radlan-dhcpcl-mib.mib

**Tables used:** rldhcpClActionStatus — 1.3.6.1.4.1.89.76.3.1.2

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.76.3.1.2.{ifIndex} i {createAndGo(4), destroy(6)}
```

### Example of obtaining an IP address via DHCP on the interface vlan

**CLI command:**

```
interface vlan 30
 ip address dhcp
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \1.3.6.1.4.1.89.76.3.1.2.100029 i 4
```

### Adding/deleting a default gateway

**MIB:** rlip.mib

**Tables used:** rllnetStaticRouteEntry — 1.3.6.1.4.1.89.26.28.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.26.28.1.4.0.0.0.0.0.1.4.{IP address}.0 i {metric(4)} \
1.3.6.1.4.1.89.26.28.1.4.0.0.0.0.0.1.4.{IP address}.0 i {remote(4)} \
1.3.6.1.4.1.89.26.28.1.4.0.0.0.0.0.1.4.{IP address}.0 i {createAndGo (4),
destroy(6)}
```

### Example of adding ip default gateway 192.168.1.10

**CLI command:**

```
ip default-gateway 192.168.1.10
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.26.28.1.7.1.4.0.0.0.0.0.1.4.192.168.1.10.0 u 4 \
1.3.6.1.4.1.89.26.28.1.8.1.4.0.0.0.0.0.1.4.192.168.1.10.0 i 4 \
1.3.6.1.4.1.89.26.28.1.10.1.4.0.0.0.0.0.1.4.192.168.1.10.0 i 4
```





## 9 CONFIGURING GREEN ETHERNET

### Global disabling of green-ethernet short-reach

**MIB:** rlgreeneth.mib

**Tables used:** rlGreenEthShortReachEnable — 1.3.6.1.4.1.89.134.2

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.134.2.0 i {true (1), false (2)}
```

#### **Example of disabling green-ethernet short-reach**

CLI command:

```
no green-ethernet short-reach
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.134.2.0 i 2
```

### Global disabling of green-ethernet energy-detect

**MIB:** rlgreeneth.mib

**Tables used:** rlGreenEthEnergyDetectEnable — 1.3.6.1.4.1.89.134.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.134.1.0 i {true (1), false (2)}
```

#### **Example of disabling green-ethernet energy-detect**

CLI command:

```
no green-ethernet energy-detect
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.134.1.0 i 2
```

### Viewing green-ethernet parameters

**MIB:** rlgreeneth.mib

**Tables used:** rlGreenEthCumulativePowerSaveMeter — 1.3.6.1.4.1.89.134.5

```
snmpwalk -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.134.5
```

#### **Example of viewing green-ethernet parameters**

CLI command:

```
show green-ethernet
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
  1.3.6.1.4.1.89.134.5
```

## 10 CONFIGURING RING PROTOCOLS

### 10.1 ERPS protocol

#### Determining the west port number

**MIB:** ELTEX-BRIDGE-ERPS-V2-MIB.mib

**Tables used:** eltexErpsMgmtRAPSWestPort — 1.3.6.1.4.1.35265.35.1.1.3.1.1.2

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.35.1.1.3.1.1.2
```

#### **Example of determining the west port number**

CLI command:

```
show erps
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.35.1.1.3.1.1.2
```

#### Viewing the state of the west port

**MIB:** ELTEX-BRIDGE-ERPS-V2-MIB.mib

**Tables used:** eltexErpsMgmtRAPSWestPortState — 1.3.6.1.4.1.35265.35.1.1.3.1.1.3

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.35.1.1.3.1.1.3
```

#### **Example of viewing the state of the west port**

CLI command:

```
show erps vlan 10
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.35.1.1.3.1.1.3
```



#### **Possible port states:**

1. Forwarding (1)
2. Blocking (2)
3. Signal-fail (3)
4. Manual-switch (4)
5. Forced-switch (5)

#### Determining the east port number

**MIB:** ELTEX-BRIDGE-ERPS-V2-MIB.mib

**Tables used:** eltexErpsMgmtRAPSEastPort — 1.3.6.1.4.1.35265.35.1.1.3.1.1.4

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.35.1.1.3.1.1.4
```

### Example of determining the east port number

CLI command:

```
show erps
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.35.1.1.3.1.1.4
```

### Viewing the east port state

**MIB:** ELTEX-BRIDGE-ERPS-V2-MIB.mib

**Tables used:** eltexErpsMgmtRAPSEastPortState — 1.3.6.1.4.1.35265.35.1.1.3.1.1.5

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.35.1.1.3.1.1.5
```

### Example of viewing the state of the east port

CLI command:

```
show erps vlan 10
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.35.1.1.3.1.1.5
```



#### **Possible port states:**

- 1. Forwarding (1)**
- 2. Blocking (2)**
- 3. Signal-fail (3)**
- 4. Manual-switch (4)**
- 5. Forced-switch (5)**

### Viewing a ring state

**MIB:** ELTEX-BRIDGE-ERPS-V2-MIB.mib

**Tables used:** eltexErpsMgmtRAPSRingState — 1.3.6.1.4.1.35265.35.1.1.3.1.1.12

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.35.1.1.3.1.1.12
```

### Example of viewing the ring state

CLI command:

```
show erps vlan 10
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.35.1.1.3.1.1.12
```



**Possible states of the erps ring:**

1. Init (1)
2. Idle (2)
3. Protection (3)
4. Manual-switch (4)
5. Forced-switch (5)
6. Pending (6)

## 10.2 Spanning-tree protocol

### Enabling/disabling Spanning-tree

**MIB:** radlan-brgmacswitch.mib

**Tables used:** rldot1dStp — 1.3.6.1.4.1.89.57.2.3

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.57.2.3.0 i {enabled(1), disabled(2)}
```

#### **Example of Spanning-tree disabling**

CLI command:

```
no spanning-tree
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.57.2.3.0 i 2
```

### Enabling/disabling spanning-tree on a configured interface

**MIB:** BRIDGE-MIB

**Tables used:** dot1dStpPortTable — 1.3.6.1.2.1.17.2.15.1.4

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.2.1.17.2.15.1.4.{ifIndex} i {enabled(1), disabled(2)}
```

#### **Example of disabling Spanning-tree on the GigabitEthernet 1/0/2 interface**

CLI command:

```
interface GigabitEthernet 1/0/2
spanning-tree disable
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.2.1.17.2.15.1.4.50 i 2
```

### Enabling/disabling BPDU packet processing by an interface with STP protocol disabled

**MIB:** radlan-bridgemibobjects-mib.mib

**Tables used:** rldot1dStpPortTable — 1.3.6.1.4.1.89.57.2.13.1.4

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.57.2.13.1.4.{ifIndex} i {filtering(1), flooding(2)}
```

### Example of BPDU filtering enabling on the GigabitEthernet 1/0/2 interface

```

CLI command:
interface gigabitEthernet 1/0/2
spanning-tree bpdu filtering

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.57.2.13.1.4.50 i 1

```

### Enabling/disabling the BPDU packet processing mode in a VLAN on which the PVST/RPVST protocol is disabled

**MIB:** eltBridgeExtMIB.mib

**Tables used:** eltBridgePvstConfigFilterBpdu — 1.3.6.1.4.1.35265.1.23.1.401.0.6.1.1.7

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.1.401.0.6.1.1.7.{VlanId} \
i {enabled(1), disabled(2)}

```

### Example of enabling BPDU filtering in VLAN 2

```

CLI command:
spanning-tree vlan 2 bpdu filtering

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.1.401.0.6.1.1.7.2 i 1

```

### Enabling/disabling the BPDU packet processing mode by an interface on which the PVST/RPVST protocol is disabled

**MIB:** eltBridgeExtMIB.mib

**Tables used:** eltBridgePvstConfigFilterBpdu — 1.3.6.1.4.1.35265.1.23.1.401.0.6.2.1.10

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.1.401.0.6.2.1.10.{VlanId}.{ifIndex} \
i {enabled(1), disabled(2)}

```

### Example of enabling BPDU filtering on the GigabitEthernet 1/0/1 interface in VLAN 2

```

CLI command:
interface gigabitEthernet 1/0/1
spanning-tree vlan 2 bpdu filtering

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.35265.1.23.1.401.0.6.2.1.10.2.49 i 1

```

### Configuring spanning-tree operation mode

**MIB:** draft-ietf-bridge-rstp.mib

**Tables used:** dot1dStpVersion — 1.3.6.1.2.1.17.2.16

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.2.1.17.2.16.0 i {stp(0), rstp(2), mstp(3)}

```

### Example of Spanning-tree operation mode setting

CLI command:

```
spanning-tree mode rstp
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.2.1.17.2.16.0 i 2
```

### Viewing port role in STP

**MIB:** radlan-bridgemibobjects-mib.mib

**Tables used:** rldot1dStpPortRole — 1.3.6.1.4.1.89.57.2.13.1.7

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.57.2.13.1.7.{ifindex}
```

### Example of viewing GigabitEthernet 0/2 role in STP

CLI command:

```
show spanning-tree GigabitEthernet 0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.57.2.13.1.7.50
```



#### **Possible port states:**

- 1. Disabled (1)**
- 2. Alternate (2)**
- 3. Backup (3)**
- 4. Root (4)**
- 5. Designated (5)**

### Viewing port state in MSTP

**MIB:** radlan-bridgemibobjects-mib.mib

**Tables used:** rldot1sMstpInstancePortState — 1.3.6.1.4.1.89.57.6.2.1.4

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.57.6.2.1.4.1.{ifindex}
```

### Example of viewing GigabitEthernet 0/2 state in MSTP

CLI command:

```
show spanning-tree GigabitEthernet 0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.57.6.2.1.4.1.50
```



#### **Possible port states:**

- 1. Disabled (1)**
- 2. Blocking (2)**
- 3. Listening (3)**
- 4. Forwarding (5)**

### Viewing the role of a port in a specific VLAN in PVST/RPVST

**MIB:** eltBridgeExtMIB.mib

**Tables used:** eltBridgePvstPortStatusPortRole — 1.3.6.1.4.1.35265.1.23.1.401.0.6.4.1.2

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.1.401.0.6.4.1.2.{VlanId}.{ifIndex}
```

#### **Example of viewing the role of the GigabitEthernet1/0/4 port in VLAN 13 in PVST/RPVST**

CLI command:

```
show spanning-tree vlan 13 GigabitEthernet1/0/4
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.1.401.0.6.4.1.2.13.52
```



#### **Possible port roles:**

- 1. Disabled (2)**
- 2. Alternate (3)**
- 3. Backup (4)**
- 4. Root (5)**
- 5. Designated (6)**

### Viewing the time from last topology change

**MIB:** BRIDGE-MIB

**Tables used:** dot1dStpTimeSinceTopologyChange — 1.3.6.1.2.1.17.2.3.0

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.17.2.3.0
```

#### **Example of viewing the time from last topology change**

CLI command:

```
show spanning-tree
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 1.3.6.1.2.1.17.2.3.0
```

### The number of topology changes

**MIB:** BRIDGE-MIB

**Tables used:** dot1dStpTopChanges — 1.3.6.1.2.1.17.2.4.0

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.17.2.4.0
```

### Example of viewing the topology change

CLI command:

```
show spanning-tree
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 1.3.6.1.2.1.17.2.4.0
```

### Viewing the interface from which the last topology change was accepted

**MIB:** eltBridgeExtMIB.mib

**Tables used:** eltdot1dStpLastTopologyChangePort — 1.3.6.1.4.1.35265.1.23.1.401.0.5.2

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.1.23.1.401.0.5.2
```

### Example of viewing the interface from which the last topology change was accepted

CLI command:

```
show spanning-tree
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 1.3.6.1.4.1.35265.1.23.1.401.0.5.2
```



## 11 MULTICAST ADDRESSING

### 11.1 Multicast addressing rules

#### Prohibition of adding the port dynamically to a multicast group

**MIB:** rlbrgmulticast.mib

**Tables used:** rIBrgStaticInetMulticastEntry — 1.3.6.1.4.1.89.116.5.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.116.5.1.6.{vlan id}.1.4.{ip address(DEC)}.1.4.0.0.0.0 x
0000000000000000 \
  1.3.6.1.4.1.89.116.5.1.7.{vlan id}.1.4.{ip address(DEC)}.1.4.0.0.0.0 x
{Interface bit mask} \
1.3.6.1.4.1.89.116.5.1.8.{vlan id}.1.4.{ip address(DEC)}.1.4.0.0.0.0 i
{createAndGo(4), destroy (6)}
```

#### **Example of prohibiting 239.200.200.17 on the GigabitEthernet 1/0/1 in vlan 622**

**CLI command:**  
interface vlan 622  
bridge multicast forbidden ip-address 239.200.200.17 add GigabitEthernet 1/0/1

**SNMP command:**  
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.116.5.1.6.622.1.4.239.200.200.17.1.4.0.0.0.0 x 0000000000000000  
\  
1.3.6.1.4.1.89.116.5.1.7.622.1.4.239.200.200.17.1.4.0.0.0.0 x 00000000000008000  
\  
1.3.6.1.4.1.89.116.5.1.8.622.1.4.239.200.200.17.1.4.0.0.0.0 i 4



**1) The total number of digits in OID 1.3.6.1.4.1.89.116.5.1.6 and OID 1.3.6.1.4.1.89.116.5.1.7 must be the same and even.**

**2) The method of calculating a bit mask can be found in the section "APPENDIX A. Bit mask calculation method".**

#### Prohibition of unregistered Multicast traffic passing

**MIB:** rlbrgmulticast.mib

**Tables used:** rIMacMulticastUnregFilterEnable — 1.3.6.1.4.1.89.55.4.1

```
snmpset -v2c -c <community> <IP address> \  

  1.3.6.1.4.1.89.55.4.1.0 x "{bit mask for interfaces}"
```

#### **Example of prohibition of unregistered Multicast traffic passing for GigabitEthernet 1/0/20-21 ports**

**CLI command:**  
interface range GigabitEthernet 1/0/20-21  
bridge multicast unregistered filtering

**SNMP command:**  
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.55.4.1.0 x "00000000000000000018"



1) To delete a setting, replace the corresponding fields by 0.

2) The method of calculating a bit mask can be found in the section "APPENDIX A. Bit mask calculation method".

### Multicast traffic filtering

**MIB:** rlbrgmulticast.mib

**Tables used:** rLMacMulticastEnable — 1.3.6.1.4.1.89.55.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.55.1.0 i {true(1), false(2)}
```

#### **Example of enabling multicast address filtering**

CLI command:

```
bridge multicast filtering
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.55.1.0 i 1
```

### Global enabling of igmp snooping

**MIB:** rlbrgmulticast.mib

**Tables used:** rLIgmpSnoopEnable — 1.3.6.1.4.1.89.55.2.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.55.2.2.0 i {true(1), false(2)}
```

#### **Example of global igmp snooping enabling**

CLI command:

```
ip igmp snooping
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.55.2.2.0 i 1
```

### Enabling igmp snooping in vlan

**MIB:** rlbrgmulticast.mib

**Tables used:** rLIgmpMldSnoopVlanEnable — 1.3.6.1.4.1.89.55.5.5.1.3

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.55.5.5.1.3.1.{vlan id} i {true(1), false(2)}
```

#### **Example of igmp snooping enabling in vlan 30**

CLI command:

```
ip igmp snooping vlan 30
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.55.5.5.1.3.1.30 i 1
```

### Viewing igmp snooping table

**MIB:** rIbrgmulticast.mib

**Tables used:** rIIgmpMldSnoopMembershipTable — 1.3.6.1.4.1.89.55.5.4

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.55.5.4
```

#### Example of viewing the igmp snooping table

CLI command:

```
show ip igmp snooping groups
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.55.5.4
```

### Multicast-tv vlan (MVR) configuration

**MIB:** rlvlan.mib

**Tables used:** vlanMulticastTvEntry — 1.3.6.1.4.1.89.48.44.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.48.44.1.1.{ifIndex} u {vlan-id} \
1.3.6.1.4.1.89.48.44.1.2.50 i {createAndGo(4), destroy(6)}
```

#### Example of configuring multicast-tv vlan 622 on the GigabitEthernet 1/0/2 interface

CLI command:

```
interface gigabitethernet 1/0/2
switchport access multicast-tv vlan 622
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.48.44.1.1.50 u 622 \
1.3.6.1.4.1.89.48.44.1.2.50 i 4
```



**Setting of multicast-tv vlan <customer/access/trunk/general> operation mode depends on the port setting mode, i.e. the switchport mode command customer/access/trunk/general.**

## 11.2 Multicast traffic restriction functions

### Multicast snooping profile creation

**MIB:** eltIpMulticast.mib

**Tables used:** eltMesIpMulticast — 1.3.6.1.4.1.35265.1.23.46.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.46.1.1.2.{Index of profile} s {profile name} \
1.3.6.1.4.1.35265. 1.23.46.1.1.3.{Index of profile} i {deny(1), permit(2)} \
1.3.6.1.4.1.35265. 1.23.46.1.1.4.{Index of profile} i {createAndGo(4),
destroy(6)}
```

### Example of creating a profile with the name IPTV (assuming the profile will have the serial number 3)

CLI command:

```
multicast snooping profile IPTV
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.46.1.1.2.3 s IPTV \
1.3.6.1.4.1.35265.1.23.46.1.1.3.3 i 1 \
1.3.6.1.4.1.35265.1.23.46.1.1.4.3 i 4
```

### Specification of Multicast address ranges in multicast snooping profile

MIB: eltlpMulticast.mib

Tables used: eltMeslpMulticast — 1.3.6.1.4.1.35265. 1.23.46.3

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265. 1.23.46.3.1.3.{index of rule}.{Index of profile} i
{ip(1),ipv6(2)} \
1.3.6.1.4.1.35265. 1.23.46.3.1.4.{index of rule}.{Index of profile} x {ip
address of the beginning of the range in hexadecimal form} \
1.3.6.1.4.1.35265. 1.23.46.3.1.5.{index of rule}.{Index of profile} x {ip
address of the end of the range in hexadecimal form} \
1.3.6.1.4.1.35265. 1.23.46.3.1.6.{index of rule}.{Index of profile} i
{createAndGo(4), destroy(6)}
```

### Example of a restriction of multicast groups 233.7.70.1-233.7.70.10 for a profile with the name IPTV (assume that the profile has a serial number 3. There are 2 rules in the first profile and one in the second)

CLI command:

```
multicast snooping profile IPTV
match ip 233.7.70.1 233.7.70.10
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.46.3.1.3.4.3 i 1 \
1.3.6.1.4.1.35265.1.23.46.3.1.4.4.3 x E9074601 \
1.3.6.1.4.1.35265.1.23.46.3.1.5.4.3 x E907460A \
1.3.6.1.4.1.35265.1.23.46.3.1.6.4.3 i 4
```



**index of rule is calculated as the sum of all rules in all profiles.**

### Assigning multicast snooping profile to a port

MIB: eltlpMulticast.mib

Tables used: eltMeslpMulticast — 1.3.6.1.4.1.35265. 1.23.46.7.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265. 1.23.46.7.1.1.{ifIndex}.{Index of profile} i {ifIndex} \
1.3.6.1.4.1.35265. 1.23.46.7.1.2.{ifIndex}.{Index of profile} i {Index of
profile} \
1.3.6.1.4.1.35265. 1.23.46.7.1.3.{ifIndex}.{Index of profile} i
{createAndGo(4), destroy(6)}
```

### Example of adding a test profile (with profile index 3) to the GigabitEthernet 1/0/2 interface

**CLI command:**

```
interface GigabitEthernet 1/0/2
 multicast snooping add test
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.46.7.1.1.50.3 i 50 \
1.3.6.1.4.1.35265.1.23.46.7.1.2.50.3 i 3 \
1.3.6.1.4.1.35265.1.23.46.7.1.3.50.3 i 4
```

### Setting a limit on the number of Multicast groups on the port

**MIB:** eltlpMulticast.mib

**Tables used:** eltMeslpMulticast — 1.3.6.1.4.1.35265.1.23.46.6.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.46.6.1.2.{ifIndex} i {MAX number}
```

### Example of setting a limit for three Multicast groups on the GigabitEthernet 1/0/2 interface

**CLI command:**

```
interface GigabitEthernet 1/0/2
 multicast snooping max-groups 3
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.46.6.1.2.50 i 3
```

## 12 CONTROL FUNCTIONS

### 12.1 AAA mechanism

#### Adding a new user

**MIB:** rlaaa.mib

**Tables used:** rIAAALocalUserTable — 1.3.6.1.4.1.89.79.17

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.79.17.1.1.{number of letters}.{Login in DEC, each letter of the
login is separated from the next one by a point} s {login} \
  1.3.6.1.4.1.89.79.17.1.2.{number of letters}.{Login in DEC, each letter of the
login is separated from the next one by a point} s "#{encoding password}" \
  1.3.6.1.4.1.89.79.17.1.3.{number of letters}.{Login in DEC, each letter of the
login is separated from the next one by a point} i {privelege level(1-15)} \
  1.3.6.1.4.1.89.79.17.1.4.{number of letters}.{Login in DEC, each letter of the
login is separated from the next one by a point} i {create and go(4)}
```

#### **Example of adding a techsup user with password 'password' and privilege level 15**

CLI command:

```
username techsup password password privilege 15
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.79.17.1.1.7.116.101.99.104.115.117.112 s techsup \
  1.3.6.1.4.1.89.79.17.1.2.7.116.101.99.104.115.117.112 s
"#5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8" \
  1.3.6.1.4.1.89.79.17.1.3.7.116.101.99.104.115.117.112 i 15
\1.3.6.1.4.1.89.79.17.1.4.7.116.101.99.104.115.117.112 i 4
```



1. The login is converted from ASCII to HEX using a table, which can be found at <https://ru.wikipedia.org/wiki/ASCII>.

2. The password is set only in encrypted form, must be written in inverted commas, and # is added before the password.

#### Enabling/disabling password format restrictions

**MIB:** rlaaa.mib

**Tables used:** rIAAAPasswordComplexityEnabled — 1.3.6.1.4.1.89.79.61

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.79.61.0 i {true (1), false(2)}
```

#### **Example of enabling a password format restriction**

CLI command:

```
passwords complexity enable
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.79.61.0 i 1
```

## Configuring authorization methods for login

MIB: rlaaa.mib

**Tables used:** rIAAAMethodListEntry — 1.3.6.1.4.1.89.79.15.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.79.15.1.2.15.{"login_c_default" in DEC, each letter of the login
is separated from the next one by a point} i
{enable(2),radius(4),tacacs(5),local(3)} \
1.3.6.1.4.1.89.79.15.1.2.15.{"login_n_default" in DEC, each login letter is
separated from the next by a dot} i {enable(2),radius(4),tacacs(5),local(3)} \
1.3.6.1.4.1.89.79.15.1.3.15.{"login_c_default" in DEC,each letter of the login
is separated from the next one by a point} i
{enable(2),radius(4),tacacs(5),local(3)} \
1.3.6.1.4.1.89.79.15.1.3.15.{"login_n_default" in DEC, each login letter is
separated from the next by a dot} i {enable(2),radius(4),tacacs(5),local(3)} \
1.3.6.1.4.1.89.79.15.1.4.15.{"login_c_default" in DEC, each letter of the login
is separated from the next one by a point} i
{enable(2),radius(4),tacacs(5),local(3)} \
1.3.6.1.4.1.89.79.15.1.4.15.{"login_n_default" in DEC, each login letter is
separated from the next by a dot} i {enable(2),radius(4),tacacs(5),local(3)} \
1.3.6.1.4.1.89.79.15.1.10.15.{"login_c_default" in DEC, each login letter is
separated from the next by a dot} i 1 \
1.3.6.1.4.1.89.79.15.1.10.15.{"login_n_default" in DEC, each login letter is
separated from the next by a dot} i 1
```

### **Example of configuring authorization methods for login**

CLI command:

```
aaa authentication login authorization default local
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.79.15.1.2.15.108.111.103.105.110.95.99.95.100.101.102.97.117.108
.116 i 3 \
1.3.6.1.4.1.89.79.15.1.2.15.108.111.103.105.110.95.110.95.100.101.102.97.117.10
8.116 i 3 \
1.3.6.1.4.1.89.79.15.1.10.15.108.111.103.105.110.95.99.95.100.101.102.97.117.10
8.116 i 1 \
1.3.6.1.4.1.89.79.15.1.10.15.108.111.103.105.110.95.110.95.100.101.102.97.117.1
08.116 i 1
```



**Field 1.3.6.1.4.1.89.79.15.1.2.15 configures the first authorization method;  
Field 1.3.6.1.4.1.89.79.15.1.3.15 configures the second authorization method;  
Field 1.3.6.1.4.1.89.79.15.1.4.15 configures the third authorization method.**



**108.111.103.105.110.95.99.95.100.101.102.97.117.108.116 is converted from the ASCII table (login\_c\_default decrypted).**

**108.111.103.105.110.95.110.95.100.101.102.97.117.108.116 is translated from the ASCII table (login\_c\_default decrypted).**

## Removing authorization method settings for login

MIB: rlaaa.mib

**Tables used:** rIAAAMethodListEntry — 1.3.6.1.4.1.89.79.15.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.79.15.1.2.15.{"login_c_default" in DEC, each login letter is
separated from the next one by a dot} i 3 \
1.3.6.1.4.1.89.79.15.1.2.15.{"login_n_default" in DEC, each login letter is
separated from the next one by a dot} i 3 \
1.3.6.1.4.1.89.79.15.1.3.15.{"login_c_default" in DEC, each login letter is
separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.3.15.{"login_n_default" in DEC, each login letter is
separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.4.15.{"login_c_default" in DEC, each login letter is
separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.4.15.{"login_n_default" in DEC, each login letter is
separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.10.15.{"login_c_default" in DEC, each login letter is
separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.10.15.{"login_n_default" in DEC, each login letter is
separated from the next one by a dot} i 0
```

### Example of removing authorization methods for login

**CLI command:**

```
no aaa authentication login default
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.79.15.1.2.15.108.111.103.105.110.95.99.95.100.101.102.97.117.108
.116 i 3 \
1.3.6.1.4.1.89.79.15.1.2.15.108.111.103.105.110.95.110.95.100.101.102.97.117.10
8.116 i 3 \
1.3.6.1.4.1.89.79.15.1.10.15.108.111.103.105.110.95.99.95.100.101.102.97.117.10
8.116 i 0 \
1.3.6.1.4.1.89.79.15.1.10.15.108.111.103.105.110.95.110.95.100.101.102.97.117.1
08.116 i 0
```



**Field 1.3.6.1.4.1.89.79.15.1.2.15 configures the first authorization method;  
Field 1.3.6.1.4.1.89.79.15.1.3.15 configures the second authorization method;  
Field 1.3.6.1.4.1.89.79.15.1.4.15 configures the third authorization method.**



**108.111.103.105.110.95.99.95.100.101.102.97.117.108.116 is converted from the ASCII table (login\_c\_default decrypted).  
108.111.103.105.110.95.110.95.100.101.102.97.117.108.116 is translated from the ASCII table (login\_c\_default decrypted).**

### Configuring authorization methods for enable

**MIB:** rlaaa.mib

**Tables used:** rIAAMethodListEntry — 1.3.6.1.4.1.89.79.15.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.79.16.1.2.16.{"enable_n_default" in DEC, each login letter is
separated from the next one by a dot} i {enable(2),radius(4),tacacs(5)} \
1.3.6.1.4.1.89.79.16.1.3.16.{"enable_c_default" in DEC, each login letter is
separated from the next one by a dot} i {enable(2),radius(4),tacacs(5)} \
1.3.6.1.4.1.89.79.16.1.4.16.{"enable_c_default" in DEC, each login letter is
separated from the next one by a dot} i {enable(2),radius(4),tacacs(5)} \
1.3.6.1.4.1.89.79.16.1.10.16.{"enable_c_default" in DEC, each letter of the
login is separated from the next one by a dot} i 1 \
1.3.6.1.4.1.89.79.16.1.10.16.{"enable_n_default" in DEC, each letter of the
login is separated from the next one by a dot} i 1
```



### Example of configuring authorization methods for enable

**CLI command:**

```
aaa authentication enable authorization default tacacs radius enable
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.79.15.1.2.16.101.110.97.98.108.101.95.99.95.100.101.102.97.117.1
08.116 i 5 \
1.3.6.1.4.1.89.79.15.1.2.16.101.110.97.98.108.101.95.110.95.100.101.102.97.117.
108.116 i 5 \
1.3.6.1.4.1.89.79.15.1.10.16.101.110.97.98.108.101.95.99.95.100.101.102.97.117.
108.116 i 1 \
1.3.6.1.4.1.89.79.15.1.10.16.101.110.97.98.108.101.95.110.95.100.101.102.97.117
.108.116 i 1
```



**Field 1.3.6.1.4.1.89.79.16.1.2.16 configures the first authorization method;  
Field 1.3.6.1.4.1.89.79.16.1.3.16 configures the second authorization method;  
Field 1.3.6.1.4.1.89.79.16.1.4.16 configures the third authorization method.**



**101.110.97.98.108.101.95.99.95.100.101.102.97.117.108.116 is converted from the ASCII table (enable\_c\_default decrypted).**

**101.110.97.98.108.101.95.110.95.100.101.102.97.117.108.116 is translated from the ASCII table (enable\_n\_default decrypted).**

### Removing authorization method settings for enable

**MIB:** rlaaa.mib

**Tables used:** rIAAAMethodListEntry — 1.3.6.1.4.1.89.79.15.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.79.15.1.2.16.{"enable_c_default" in DEC, each letter of the
login is separated from the next one by a dot} i 2 \
1.3.6.1.4.1.89.79.15.1.2.16.{"enable_n_default" in DEC, each letter of the
login is separated from the next one by a dot} i 2 \
1.3.6.1.4.1.89.79.15.1.3.16.{"enable_c_default" in DEC, each letter of the
login is separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.3.16.{"enable_n_default" in DEC, each letter of the
login is separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.4.16.{"enable_c_default" in DEC, each letter of the
login is separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.4.16.{"enable_n_default" in DEC, each letter of the
login is separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.10.16.{"enable_c_default" in DEC, each letter of the
login is separated from the next one by a dot} i 0 \
1.3.6.1.4.1.89.79.15.1.10.16.{"enable_n_default" in DEC, each letter of the
login is separated from the next one by a dot} i 0
```

### Example of removing authorization methods for enable

**CLI command:**

```
no aaa authentication enable default
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.79.15.1.2.16.101.110.97.98.108.101.95.99.95.100.101.102.97.117.1
08.116 i 2 \
1.3.6.1.4.1.89.79.15.1.2.16.101.110.97.98.108.101.95.110.95.100.101.102.97.117.
108.116 i 2 \
```

```
1.3.6.1.4.1.89.79.15.1.10.16.101.110.97.98.108.101.95.99.95.100.101.102.97.117.
108.116 i 0 \
1.3.6.1.4.1.89.79.15.1.10.16.101.110.97.98.108.101.95.110.95.100.101.102.97.117
.108.116 i 0
```



**Field 1.3.6.1.4.1.89.79.15.1.2.15** configures the first authorization method;  
**Field 1.3.6.1.4.1.89.79.15.1.3.15** configures the second authorization method;  
**Field 1.3.6.1.4.1.89.79.15.1.4.15** configures the third authorization method.



**101.110.97.98.108.101.95.99.95.100.101.102.97.117.108.116** is converted from the ASCII table (`enable_c_default` decrypted).

**101.110.97.98.108.101.95.110.95.100.101.102.97.117.108.116** is translated from the ASCII table (`enable_n_default` decrypted).

## 12.2 Access configuration

### Enabling the Telnet server

**MIB:** radlan-telnet-mib.mib

**Tables used:** rITelnetEnable — 1.3.6.1.4.1.89.58.7

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.58.7.0 i {on(1), off(2)}
```

#### Example of Telnet server enabling

CLI command:  
`ip telnet server`

SNMP command:  

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.58.7.0 i 1
```

### Enabling the SSH server

**MIB:** rlssh.mib

**Tables used:** rISshServerEnable — 1.3.6.1.4.1.89.78.2.102

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.78.2.102.0 i {on(1), off(2)}
```

#### Example of enabling the SSH server

CLI command:  
`ip ssh server`

SNMP command:  

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.78.2.102.0 i 1
```

---

### Viewing active sessions

**MIB:** rIAAA.mib

**Tables used:** rIAAAUserInetName — 1.3.6.1.4.1.89.79.57.1.5

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.79.57.1.5
```

#### **Example of viewing active sessions**

CLI command:

```
Show users
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.79.57.1.5
```

## 13 PORT MIRRORING

### Port Mirroring configuration

**MIB:** rfc2613.mib

**Tables used:** portCopyTable — 1.3.6.1.2.1.16.22.1.3.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.2.1.16.22.1.3.1.1.4.{ifindex src port}.{ifindex dst port} i
{copyRxOnly(1), copyTxOnly(2), copyBoth(3)} \
  1.3.6.1.2.1.16.22.1.3.1.1.5.{ifindex src port}.{ifindex dst port} i
{createAndGo(4), destroy(6)}
```

### **Example of traffic mirroring from GigabitEthernet 1/0/1 to GigabitEthernet 1/0/2**

**CLI command:**

```
interface GigabitEthernet 1/0/2
  port monitor GigabitEthernet 1/0/1
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.2.1.16.22.1.3.1.1.4.49.50 i 3 \
  1.3.6.1.2.1.16.22.1.3.1.1.5.49.50 i 4
```

### Enabling vlan mirroring

**MIB:** rfc2613.mib

**Tables used:** portCopyTable — 1.3.6.1.2.1.16.22.1.3.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.2.1.16.22.1.3.1.1.4.{ifindex vlan}.{ifindex dst port} i
{copyRxOnly(1)} \
  1.3.6.1.2.1.16.22.1.3.1.1.5.{ifindex vlan}.{ifindex dst port} i
{createAndGo(4), destroy(6)}
```

### **Example of configuring vlan 622 mirroring on the GigabitEthernet 1/0/2 interface**

**CLI command:**

```
interface GigabitEthernet 1/0/2
  port monitor vlan 622
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.2.1.16.22.1.3.1.1.4.100621.50 i 1 \
  1.3.6.1.2.1.16.22.1.3.1.1.5.100621.50 i 4
```

### Configuring the mirroring mode

**MIB:** RADLAN-SMON-MIB

**Tables used:** rIPortCopyMode — 1.3.6.1.4.1.89.84.4

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.84.4.0 i {monitor-only(1), network(2)}
```

### Example of setting the mirroring mode

**CLI command:**

```
port monitor mode network
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.84.4.0 i 2
```

### Assigning VLANs for remote monitoring

**MIB:** ELTEX-MES-SMON-MIB

**Tables used:** eltPortCopyRemoteTable — 1.3.6.1.4.1.35265.1.23.84.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.84.1.1.2.1 i {vlan id} \
1.3.6.1.4.1.35265.1.23.84.1.1.2.2 i {vlan id} \
1.3.6.1.4.1.35265.1.23.84.1.1.4.1 i {createAndGo(4), destroy(6)}
1.3.6.1.4.1.35265.1.23.84.1.1.4.2 i { createAndGo(4), destroy(6)}
```

### Example of assigning vlan 100 for remote monitoring

**CLI command:**

```
port monitor remote vlan 100
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.84.1.1.2.1 i 100 \
1.3.6.1.4.1.35265.1.23.84.1.1.2.2 i 100 \
1.3.6.1.4.1.35265.1.23.84.1.1.4.1 i 4 \
1.3.6.1.4.1.35265.1.23.84.1.1.4.2 i 4
```

### Enabling Remote Mirroring (RSPAN)

**MIB:** RADLAN-SMON-MIB

**Tables used:** rIPortCopyVlanTaggingTable — 1.3.6.1.4.1.89.84.3

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.84.3.1.1.{ifindex} i {true}1, false(2)}
```

### Example of enabling remote mirroring

**CLI command:**

```
interface GigabitEthernet 0/2
port monitor remote
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.84.3.1.1.50 i 1
```

### Setting limits (descriptors) for mirrored traffic transmission

**MIB:** ELTEX-MES-QOS-TAIL-DROP-MIB

**Tables used:** eltQoS TailDropMirrorRxDescriptorsLimit — 1.3.6.1.4.1.35265.1.23.12.1.1.5

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.12.1.1.5.0 i {rx mirror descriptors limit} \
1.3.6.1.4.1.35265.1.23.12.1.1.6.0 i {tx mirror descriptors limit}
```

### Example of setting limits

CLI command:

```
qos tail-drop mirror-limit rx 2650  
qos tail-drop mirror-limit tx 2650
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.12.1.1.5.0 i 2650 \  
1.3.6.1.4.1.35265.1.23.12.1.1.6.0 i 2650
```

## 14 PHYSICAL LAYER DIAGNOSTIC FUNCTIONS

### 14.1 Copper-wire cable diagnostics

#### Running the TDR test for the port

**MIB:** rlphy.mib

**Tables used:** rlPhyTestSetType — 1.3.6.1.4.1.89.90.1.1.1.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.90.1.1.1.1.{ifIndex} i 2
```

#### **Example of tdr launch for GigabitEthernet port 1/0/12**

CLI command:

```
test cable-diagnostics tdr interface GigabitEthernet 1/0/12
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.90.1.1.1.1.60 i 2
```



**To run the tdr-fast test, specify the i 25 parameter.**

#### Reading information on pairs during TDR testing

**MIB:** eltPhy.mib

**Tables used:** eltPhyTdrTestTable — 1.3.6.1.4.1.35265.1.23.90.1.1

- 1 (1-2) pair status:

```
snmpwalk -v2c -c <community> <IP address> \
  1.3.6.1.4.1.35265. 1.23.90.1.1.1.2.{ifIndex}
```

- 2 (3-6) pair status:

```
snmpwalk -v2c -c <community> <IP address> \
  1.3.6.1.4.1.35265. 1.23.90.1.1.1.3.{ifIndex}
```

- 3 (4-5) pair status:

```
snmpwalk -v2c -c <community> <IP address> \
  1.3.6.1.4.1.35265. 1.23.90.1.1.1.4.{ifIndex}
```

- 4 (7-8) pair status:

```
snmpwalk -v2c -c <community> <IP address> \
  1.3.6.1.4.1.35265. 1.23.90.1.1.1.5.{ifIndex}
```

### Example of viewing the status of pair 1 on the GigabitEthernet 1/0/12 interface

CLI command:

```
show cable-diagnostics tdr interface GigabitEthernet 1/0/12
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.90.1.1.1.2.60
```



#### Pair statuses:

**test-failed(0)** — a physical fault; or at the time of the request, a line diagnostics is performed;

**ok(1)** — a pair is fine;

**open(2)** — breakup;

**short(3)** — pair contacts are shortened;

**impedance-mismatch(4)** — the difference in resistance (too much attenuation in a line);

**short-with-pair-1(5)** — short-circuit between pairs;

**short-with-pair-2(6)** — short-circuit between pairs;

**short-with-pair-3(7)** — short-circuit between pairs;

**short-with-pair-4(8)** — short-circuit between pairs.

### Measurement of pairs' length for the TDR method

**MIB:** eltPhy.mib

**Tables used:** eltPhyTdrTestTable — 1.3.6.1.4.1.35265. 1.23.90.1.1

- 1 (1-2) pair length:

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265. 1.23.90.1.1.1.6.{ifIndex}
```

- 2 (3-6) pair length:

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265. 1.23.90.1.1.1.7.{ifIndex}
```

- 3 (4-5) length:

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265. 1.23.90.1.1.1.8.{ifIndex}
```

- 4 (7-8) pair length:

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265. 1.23.90.1.1.1.9.{ifIndex}
```

### Example of pair 4 length measurement for the tdr method on the GigabitEthernet 1/0/12 interface

CLI command:

```
show cable-diagnostics tdr interface GigabitEthernet 1/0/12
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.90.1.1.1.9.60
```



## Measurement of cable length using the attenuation-based method

**MIB:** rlphy.mib

**Tables used:** rlPhyTestGetResult — 1.3.6.1.4.1.89.90.1.2.1.3

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.90.1.2.1.3.{ifIndex}
```

### Example of cable length measurement on all active ports

CLI command:

```
show cable-diagnostics cable-length
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.90.1.2.1.3
```

## 14.2 Optical transceiver diagnostics

### DDM readings

**MIB:** rlphy.mib

**Tables used:** rlPhyTestGetResult — 1.3.6.1.4.1.89.90.1.2.1.3

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.90.1.2.1.3.{port index}.{parameter type}
```

### Example of requesting DDM readings from the TenGigabitEthernet 1/0/1 interface (for all parameters)

CLI command:

```
show fiber-ports optical-transceiver interface TenGigabitEthernet 0/1
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.90.1.2.1.3.105
```



**Parameter type can take the following values:**

- rlPhyTestTableTransceiverTemp (5)** — SFP transceiver temperature;
- rlPhyTestTableTransceiverSupply (6)** — power supply voltage in  $\mu\text{V}$ ;
- rlPhyTestTableTxBias (7)** — bias current in  $\mu\text{A}$ ;
- rlPhyTestTableTxOutput (8)** — transmission power level in mDbm;
- rlPhyTestTableRxOpticalPower (9)** — reception power level in mDbm.

### Viewing information about the SFP transceiver

**MIB:** eltMes.mib

**Tables used:** eltMesPhdTransceiver — 1.3.6.1.4.1.35265.1.23.53

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.53.1.1.1.{parameter type}.{port index}
```

### Example of viewing the SFP serial number from the GigabitEthernet 1/0/2 interface

CLI command:

```
show fiber-ports optical-transceiver interface GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.53.1.1.1.6.50
```



Parameter type can take the following values:

**eltPhdTransceiverInfoConnectorType (1)** — type of SFP transceiver connector;

The most common values are:

0 — unknown;

1 — SC;

7 — LC;

34 — RJ-45.

**eltPhdTransceiverInfoType (2)** — type of SFP transceiver;

The most common values are:

0 — unknown;

3 — SFP/SFP+.

**eltPhdTransceiverInfoComplianceCode (3)** — SFP transceiver compliance code;

**eltPhdTransceiverInfoWaveLength (4)** — wavelength in nm;

**eltPhdTransceiverInfoVendorName (5)** — name of the vendor of the SFP transceiver;

**eltPhdTransceiverInfoSerialNumber (6)** — serial number of the SFP transceiver;

**eltPhdTransceiverInfoFiberDiameterType (7)** — diameter of the fiber core.

Values:

1 — 9 microns;

2 — 50 microns;

3 — 62.5 microns;

4 — copper cable;

65635 — unknown.

**eltPhdTransceiverInfoTransferDistance (8)** — maximum transmission line length of the SFP transceiver in m;

**eltPhdTransceiverInfoDiagnostic (9)** — DDM support by SFP transceiver.

Values:

1 — true;

2 — false.

## 15 POWER OVER ETHERNET (POE)

### Viewing PoE power consumption/nominal capacity

**MIB:** rfc3621.mib

**Tables used:** pethMainPseEntry — 1.3.6.1.2.1.105.1.3.1.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.105.1.3.1.1.{nominal(2), consumed(4)}.{unit}
```

#### **Example of viewing power consumption**

CLI command:

```
show power inline
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.105.1.3.1.1.4.1
```

### Viewing the PoE temperature sensor readings

**MIB:** rIPoe.mib

**Tables used:** rIPethPowerPseTemperatureSensor — 1.3.6.1.4.1.89.108.3.1.6

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.108.3.1.6.{unit}
```

#### **Example of viewing a temperature sensor readings**

CLI command:

```
show power inline
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.108.3.1.6.1
```

### Viewing power limit on the PoE interface

**MIB:** rIPoe.mib

**Tables used:** rlpethPsePortOperPowerLimit — 1.3.6.1.4.1.89.108.1.1.9

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.108.1.1.9.{unit}.{ifindex}
```

#### **Example of viewing the power limit on the GigabitEthernet 1/0/2 interface**

CLI command:

```
show power inline GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.108.1.1.9.1.50
```

### Viewing power value on the PoE interface

**MIB:** rfc3621.mib

**Tables used:** pethPsePortActualPower — 1.3.6.1.2.1.105.1.1.1.15

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.105.1.1.1.15.{unit}.{ifindex}
```

#### **Example of viewing the power value on the GigabitEthernet 1/0/2 interface**

CLI command:

```
show power inline GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.105.1.1.1.15.1.50
```

### View current value on the PoE interface

**MIB:** rIPoe.mib

**Tables used:** rlpethPsePortOutputCurrent — 1.3.6.1.4.1.89.108.1.1.4

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.108.1.1.4.{unit}.{ifindex}
```

#### **Example of viewing the current value on the GigabitEthernet 1/0/2 interface**

CLI command:

```
show power inline GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.108.1.1.4.1.50
```

### Viewing voltage value on the PoE interface

**MIB:** rIPoe.mib

**Tables used:** rlpethPsePortOutputVoltage — 1.3.6.1.4.1.89.108.1.1.3

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.108.1.1.3.{unit}.{ifindex}
```

#### **Example of viewing the voltage value on the GigabitEthernet 1/0/2 interface**

CLI command:

```
show power inline GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.108.1.1.3.1.50
```

---

### Disabling Power over Ethernet on the port

**MIB:** rfc3621.mib

**Tables used:** pethPsePortAdminEnable — 1.3.6.1.2.1.105.1.1.1.3

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.2.1.105.1.1.1.3.{unit}.{ifindex} i {auto(1), never(2)}
```

#### **Example of disabling PoE on the GigabitEthernet 1/0/2 interface**

CLI command:

```
interface GigabitEthernet 1/0/2  
power inline never
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.2.1.105.1.1.1.3.1.50 i 2
```

## 16 SECURITY FUNCTIONS

### 16.1 Port security functions

#### Limiting the number of MAC addresses learned on Ethernet ports

**MIB:** rllInterfaces.mib

**Tables used:** swIfTable — 1.3.6.1.4.1.89.43.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.43.1.1.38.{ifIndex} i {max mac addresses}
```

#### **Example of a limit of 20 MAC addresses on the GigabitEthernet 1/0/2 port**

CLI command:

```
interface GigabitEthernet 1/0/2
  port security max 20
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.43.1.1.38.50 i 20
```

#### Enabling port security

**MIB:** rllInterfaces.mib

**Tables used:**swIfPortLockIfRangeTable — 1.3.6.1.4.1.89.43.6

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.43.6.1.3.1 i {locked(1), unlocked(2)} \
  1.3.6.1.4.1.89.43.6.1.4.1 i {discard(1), forwardNormal(2), discardDisable(3),
action on a package that is not covered by port security regulations} \
  1.3.6.1.4.1.89.43.6.1.5.1 i {true(1), false(2). For trap sending} \
  1.3.6.1.4.1.89.43.6.1.6.1 i {trap sending frequency (s)} \
  1.3.6.1.4.1.89.43.6.1.2.1 x {ifindex as a bit mask}
```

#### **Example of configuring port security for GigabitEthernet 1/0/1-2 interfaces**

CLI command:

```
interface range GigabitEthernet 1/0/1-2
  port security discard trap 30
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.43.6.1.3.1 i 1 \
  1.3.6.1.4.1.89.43.6.1.4.1 i 1 \
  1.3.6.1.4.1.89.43.6.1.5.1 i 1 \
  1.3.6.1.4.1.89.43.6.1.6.1 i 30 \
  1.3.6.1.4.1.89.43.6.1.2.1 x "0000000000000C0"
```



**Example of bit mask calculation is given in section "APPENDIX A. Bit mask calculation method".**

### Setting port security operation mode

**MIB:** rllInterfaces.mib

**Tables used:** swlfTable — 1.3.6.1.4.1.89.43.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.43.1.1.37.{ifIndex} i {disabled(1), dynamic(2), secure-
  permanent(3), secure-delete-on-reset(4)}
```

#### **Example of setting a limit on the number of MAC addresses learned on the GigabitEthernet 1/0/2 port**

```
CLI command:
interface GigabitEthernet 1/0/2
  port security mode max-addresses
```

```
SNMP command:
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.43.1.1.37.50 i 2
```

### Viewing port security status

**MIB:** rllInterfaces.mib

**Tables used:** swlfLockAdminStatus — 1.3.6.1.4.1.89.43.1.1.8

```
snmpwalk -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.43.1.1.8
```

#### **Example of viewing port security state**

```
CLI command:
show ports security
```

```
SNMP command:
snmpwalk -v2c -c public 192.168.1.30 \
  1.3.6.1.4.1.89.43.1.1.8
```

### Viewing port security type

**MIB:** rllInterfaces.mib

**Tables used:** swlfAdminLockAction — 1.3.6.1.4.1.89.43.1.1.20

```
snmpwalk -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.43.1.1.20
```

#### **Example of viewing port security type**

```
CLI command:
show ports security
```

```
SNMP command:
snmpwalk -v2c -c public 192.168.1.30 \
  1.3.6.1.4.1.89.43.1.1.20
```

### Viewing the maximum specified number of MAC addresses learned on Ethernet ports

**MIB:** rllInterfaces.mib

**Tables used:** swifLockMaxMacAddresses — 1.3.6.1.4.1.89.43.1.1.38

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.43.1.1.38
```

#### **Example of viewing the maximum specified number of MAC addresses learned on Ethernet ports**

CLI command:

```
show ports security
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.43.1.1.38
```

#### **Switching the port to isolation mode**

**MIB:** rlprotectedport.mib

**Tables used:** rlProtectedPortsTable — 1.3.6.1.4.1.89.132.1

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.132.1.1.1.{Ifindex} i {not-protected(1), protected(2)}
```

#### **Example of isolation settings on GigabitEthernet 1/0/1 and GigabitEthernet 1/0/2 ports**

CLI command:

```
interface range GigabitEthernet 1/0/1-2  
switchport protected-port
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.132.1.1.1.49 i 2 \  
1.3.6.1.4.1.89.132.1.1.1.50 i 2
```

#### **Setting up sending traffic to uplink-port**

**MIB:** RADLAN-vlan-MIB

**Tables used:** vlanPrivateEdgeStatus — 1.3.6.1.4.1.89.48.37.1.2

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.48.37.1.1.{Ifindex} i {ifindex} \  
1.3.6.1.4.1.89.48.37.1.2.{Ifindex} i {createandGo(4), destroy(6)}
```

#### **Example of setting up sending traffic to the uplink port**

CLI command:

```
interface GigabitEthernet 1/0/6  
switchport protected GigabitEthernet 1/0/8
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.48.37.1.1.54 i 56 \  
1.3.6.1.4.1.89.48.37.1.2.54 i 4
```

#### **Create a static bind in MAC table**

**MIB:** Q-BRIDGE-MIB



### Tables used: dot1qStaticUnicastTable — 1.3.6.1.2.1.17.7.1.3.1

```
snmpset -v2c -c <community> -t 20 -r 0 <IP address> \
1.3.6.1.2.1.17.7.1.3.1.1.4.{vlan id}.{mac address(DEC)}. MAC address bytes are
separated by points}.{ifIndex} i {other(1), invalid(2), permanent(3),
deleteOnReset(4), deleteOnTimeout(5)}
```

### Example of binding MAC address 00:22:68:7d:0f:3f in vlan 622 to the GigabitEthernet 1/0/2 interface in the secure mode (by default, the permanent mode is used)

#### CLI command:

```
mac address-table static 00:22:68:7d:0f:3f vlan 622 interface GigabitEthernet
1/0/2 secure
```

#### SNMP command:

```
snmpset -v2c -c private -t 20 -r 0 192.168.1.30 \
1.3.6.1.2.1.17.7.1.3.1.1.4.622.0.34.104.125.15.63.50 i 1
```

### View MAC table

#### MIB: Q-BRIDGE-MIB

### Tables used: dot1qTpFdbTable — 1.3.6.1.2.1.17.7.1.2.2

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.17.7.1.2.2
```

### Example of viewing a MAC table

#### CLI command:

```
show mac address-table
```

#### SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.2.1.17.7.1.2.2
```

### Creating a static bind in arp table

#### MIB: RFC1213-MIB

### Tables used: ipNetToMediaTable — 1.3.6.1.2.1.4.22

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.2.1.4.22.1.2.{vlan id}.{IP address} x {„MAC address"} \
1.3.6.1.2.1.4.22.1.3.{vlan id}.{IP address} a {IP address} \
1.3.6.1.2.1.4.22.1.4.{vlan id}.{IP address} i 4
```

### Example of binding ip 192.168.1.21 and MAC aa:bb:cc:dd:ee:ff to vlan 1

#### CLI command:

```
arp 192.168.1.21 aa:bb:cc:dd:ee:ff vlan 1
```

#### SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.2.1.4.22.1.2.100000.192.168.1.21 x "aabbccddeeff" \
1.3.6.1.2.1.4.22.1.3.100000.192.168.1.21 a 192.168.1.21 \
1.3.6.1.2.1.4.22.1.4.100000.192.168.1.21 i 4
```



1. To remove the binding, assign the value in the field 1.3.6.1.2.1.4.22.1.4.
2. The IP address of the device and the IP address of the created static record in the arp table must be in the same subnet.

### View ARP table

**MIB:** RFC1213-MIB.mib, Q-BRIDGE-MIB.mib

**Tables used:**

pNetToMediaPhysAddress — 1.3.6.1.2.1.4.22.1.2

dot1qTpFdbEntry — 1.3.6.1.2.1.17.7.1.2.2.1

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.4.22.1.2.{(2) ip address, (3)MAC address}
```

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.2.1.17.7.1.2.2.1
```

### **Example of viewing ARP table**

CLI command:

```
show arp
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.4.22.1.2  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.17.7.1.2.2.1
```



1. The pNetToMediaPhysAddress table value shows vlan IP and MAC addresses.
2. The dot1qTpFdbEntry table value shows the status and the identification number of the port from which the device is available.

## **16.2 DHCP control and option 82**

### Enabling/disabling the DHCP server function on the switch

**MIB:** rldhcp.mib

**Tables used:** rIDhcpRelayInterfaceListTable — 1.3.6.1.4.1.89.38.29

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.38.30.0 i {true(1), false(2)}
```

### **Example of enabling DHCP server on the switch**

CLI command:

```
ip dhcp server
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.38.30.0 i 1
```

### Viewing the dhcp snooping table entries

**MIB:** rIBridgeSecurity.mib

**Tables used:** rIIPDhcpSnoopEntry — 1.3.6.1.4.1.89.112.1.11.1

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.112.1.11.1
```

#### **Example of viewing the dhcp snooping table**

CLI command:

```
Show ip dhcp snooping binding
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.112.1.11.1
```

### Enabling/disabling DHCP/DHCPv6 snooping globally

**MIB:** rlbridge-security.mib

**Tables used:** rIIPDhcpSnoopEnable — 1.3.6.1.4.1.89.112.1.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.112.1.2.0 i {enable(1), disable(2)}
```

#### **Example of enabling global dhcp snooping**

CLI command:

```
ip dhcp snooping
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.112.1.2.0 i 1
```

### Enabling/disabling dhcp snooping in vlan

**MIB:** rlbridge-security.mib

**Tables used::** rIIPDhcpSnoopEnableVlanTable — 1.3.6.1.4.1.89.112.1.12

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.112.1.12.1.2.{vlan id} i {createAndGo(4), destroy(6)}
```

#### **Example of adding dhcp snooping to vlan 622**

CLI command:

```
ip dhcp snooping vlan 622
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.112.1.12.1.2.622 i 4
```

### Configuring ip DHCP information option

**MIB:** rlbridgesecurity.mib

**Tables used:** rllpDhcpOpt82InsertionEnable — 1.3.6.1.4.1.89.112.1.8

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.112.1.8.0 i {enable(1), disable(2)}
```

#### **Example of configuring ip DHCP information option**

CLI command:

```
ip dhcp information option
```

SNMP command:

```
snmpset -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.112.1.8.0 i 1
```

### Configuring a dhcp trusted port

**MIB:** rlbridge-security.mib

**Tables used:** rllpDhcpSnoopTrustedPortTable — 1.3.6.1.4.1.89.112.1.13

```
snmpset -v2c -c <community> <IP address>  
1.3.6.1.4.1.89.112.1.13.1.2.{ifIndex} i {createAndGo(4), destroy(6)}
```

#### **Example of configuring the GigabitEthernet 1/0/2 trusted interface**

CLI command:

```
interface GigabitEthernet 1/0/2  
ip dhcp snooping trust
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.112.1.13.1.2.50 i 4
```

### Configuring DHCP relay on vlan

**MIB:** rldhcp.mib

**Tables used:**

rldhcpRelayInterfaceListVlanId1To1024 — 1.3.6.1.4.1.89.38.29.1.3  
rldhcpRelayInterfaceListVlanId1025To2048 — 1.3.6.1.4.1.89.38.29.1.4  
rldhcpRelayInterfaceListVlanId2049To3072 — 1.3.6.1.4.1.89.38.29.1.5  
rldhcpRelayInterfaceListVlanId3073To4094 — 1.3.6.1.4.1.89.38.29.1.6

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.38.29.1.3.1 x {bit mask}
```

#### **Example of configuring ip DHCP relay enable on vlan 1**

CLI command:

```
Interface vlan 1  
Ip dhcp relay enable
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.38.29.1.3.1 x 800000000000
```

### Example of configuring ip DHCP relay enable on vlan 1026

**CLI command:**

```
Interface vlan 1026
Ip dhcp relay enable
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.38.29.1.4.1 x 400000000000
```



An example of calculating a bit mask can be found in section "APPENDIX A. Bit mask calculation method".

## 16.3 IP-source Guard

### Enabling/disabling ip source guard globally

**MIB:** rlbridge-security.mib

**Tables used:** rllpSourceGuardEnable — 1.3.6.1.4.1.89.112.2.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.112.2.2.0 i {enable(1), disable(2)}
```

### Example of enabling ip source guard globally

**CLI command:**

```
ip source-guard
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.112.2.2.0 i 1
```

### Creating ip source guard static bind

**MIB:** rlbridge-security.mib

**Tables used:** rllpDhcpSnoopStaticTable — 1.3.6.1.4.1.89.112.1.10

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.112.1.10.1.3.{vlan id}.{MAC in DEC. Each MAC address byte is
separated from a previous one by a point} a {ip address (DEC)} \
1.3.6.1.4.1.89.112.1.10.1.4.{vlan id}.{MAC in DEC. Each MAC address byte is
separated from a previous one by a point} i {ifIndex} \
1.3.6.1.4.1.89.112.1.10.1.5.{vlan id}.{MAC in DEC. Each MAC address byte is
separated from a previous one by a point} i {createAndGo(4), destroy(6)}
```

### Example of binding MAC address 00:11:22:33:44:55 to IP 192.168.1.34, vlan 622, interface GigabitEthernet 1/0/9

**CLI command:**

```
ip source-guard binding 00:11:22:33:44:55 622 192.168.1.34 GigabitEthernet 1/0/9
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.112.1.10.1.3.622.0.17.34.51.68.85 a 192.168.1.34 \
1.3.6.1.4.1.89.112.1.10.1.4.622.0.17.34.51.68.85 i 57 \
1.3.6.1.4.1.89.112.1.10.1.5.622.0.17.34.51.68.85 i 4
```



### Example of enabling arp inspection globally

```
CLI command:
ip arp inspection

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.112.3.2.0 i 1
```

### Enabling/disabling arp inspection on vlan

**MIB:** rIbridge-security.mib

**Tables used:** rIIpArpInspectEnableVlanTable — 1.3.6.1.4.1.89.112.3.6

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.112.3.6.1.3.{vlan id} i {createAndGo(4), destroy(6)}
```

### Example of enabling arp inspection on vlan 622

```
CLI command:
ip arp inspection vlan 622

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.112.3.6.1.3.622 i 4
```

### Configuring an arp inspection trusted interface

**MIB:** rIbridge-security.mib

**Tables used:** rIIpArpInspectTrustedPortRowStatus — 1.3.6.1.4.1.89.112.3.7.1.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.112.3.7.1.2.{ifIndex} i {createAndGo(4), destroy(6)}
```

### Example of configuring the GigabitEthernet 1/0/2 trusted interface

```
CLI command:
interface GigabitEthernet 1/0/2
 ip arp inspection trust

SNMP command:
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.112.3.7.1.2.50 i 4
```

### Binding ip arp inspection to vlan

**MIB:** rIbridge-security.mib

**Tables used:** rIIpArpInspectAssignedListName — 1.3.6.1.4.1.89.112.3.6.1.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.112.3.6.1.2.{vlan id} s {list name}
```

### Example of binding the test list to vlan 622

CLI command:

```
ip arp inspection list assign 100 test
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.112.3.6.1.2.622 s test
```

## 16.5 Port-based client authentication (802.1x)

### Enabling 802.1x authentication on the switch

**MIB:** dot1xPaeSystem.mib

**Tables used:** dot1xPaeSystemAuthControl — 1.0.8802.1.1.1.1.1.1

```
snmpset -v2c -c <community> <IP address> \  
1.0.8802.1.1.1.1.1.1.0 i {enabled(1), disabled(2)}
```

### Example of enabling 802.1x

CLI Command:

```
dot1x system-auth-control
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.0.8802.1.1.1.1.1.1.0 i 1
```

### Enabling periodic re-authentication of the client

**MIB:** draft-ietf-bridge-8021x.mib

**Tables used:** dot1xAuthReAuthEnabled — 1.0.8802.1.1.1.1.2.1.1.13

```
snmpset -v2c -c <community> <IP address> \  
1.0.8802.1.1.1.1.2.1.1.13.{ifIndex} i {true(1), false(2)}
```

### Example of enabling periodic re-authentication of the client on the GigabitEthernet 1/0/2 interface

CLI Command:

```
interface gigabitethernet 1/0/2  
dot1x reauthentication
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.0.8802.1.1.1.1.2.1.1.13.50 i 1
```



### Period between re-authentications configuration

**MIB:** draft-ietf-bridge-8021x.mib

**Tables used:** dot1xAuthConfigTable — 1.0.8802.1.1.1.1.2.1.1.12

```
snmpset -v2c -c <community> <IP address> \  
1.0.8802.1.1.1.1.2.1.1.12.{ifIndex} u {size 300-4294967295}
```

### **Example of setting a period of 300 seconds between repeated checks on the interface GigabitEthernet 1/0/2**

**CLI Command:**

```
interface gigabitEthernet 1/0/2  
dot1x timeout reauth-period 300
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.0.8802.1.1.1.1.2.1.1.12.50 u 300
```

### Configuring 802.1x authentication modes on the interface

**MIB:** draft-ietf-bridge-8021x.mib

**Tables used:** dot1xAuthConfigTable — 1.0.8802.1.1.1.1.2.1.1.6

```
snmpset -v2c -c <community> <IP address> \  
1.0.8802.1.1.1.1.2.1.1.6.{ifIndex} i {force-Unauthorized(1), auto(2), force-  
Authorized(3)}
```

### **Example of 802.1X authentication configuration in auto mode on the GigabitEthernet 1/0/2 interface**

**CLI Command:**

```
interface gigabitEthernet 1/0/2  
dot1x port-control auto
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.0.8802.1.1.1.1.2.1.1.6.50 i 2
```

### Enabling authentication based on users' MAC addresses

**MIB:** radlan-dot1x-mib.mib

**Tables used:** rldot1xAuthenticationPortTable — 1.3.6.1.4.1.89.95.10.1.1

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.95.10.1.1.{ifIndex} i {destroy(1), mac-and-802.1x(2), mac-  
only(3)}
```

### **Example of enabling MAC-based authentication on the GigabitEthernet 1/0/3 interface**

**CLI Command:**

```
interface gigabitEthernet 1/0/3  
dot1x authentication mac
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.95.10.1.1.51 i 3
```

**Allowing the presence of one or more clients on an authorized 802.1x port****MIB:** rlinterfaces.mib**Tables used:** swIfTable — 1.3.6.1.4.1.89.43.1.1.30

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.43.1.1.30.{ifIndex} i {single(1), none(2), multi-sessions(3)}
```

**Example of multiple client permissions on the GigabitEthernet 1/0/3 interface****CLI Command:**

```
interface GigabitEthernet 1/0/3  
dot1x host-mode multi-sessions
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.43.1.1.30.51 i 3
```

**Enabling one or two authentication, authorization and accounting (AAA) methods for use on IEEE 802.1x interfaces****MIB:** rIAAA.mib**Tables used:** rIAAAEapMethodListTable — 1.3.6.1.4.1.89.97.1

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.97.1.1.1.7.{"default" in DEC, each letter is separated from the  
next one by a point} s {authentication list}  
\1.3.6.1.4.1.89.97.1.1.2.7.{"default" in DEC, each letter is separated from the  
next one by a point} i {Deny(0), radius(1), none(2)} \  
1.3.6.1.4.1.89.97.1.1.3.7.{"default" in DEC, each letter is separated from the  
next one by a point} i {Deny(0), radius(1), none(2)} \  
1.3.6.1.4.1.89.97.1.1.7.7.{"default" in DEC, each letter is separated from the  
next one by a point} i 1
```

**Example of enabling RADIUS server list for user authentication****CLI Command:**

```
aaa authentication dot1x default radius none
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.97.1.1.1.7.100.101.102.97.117.108.116 s default \  
1.3.6.1.4.1.89.97.1.1.2.7.100.101.102.97.117.108.116 i 1 \  
1.3.6.1.4.1.89.97.1.1.3.7.100.101.102.97.117.108.116 i 2 \  
1.3.6.1.4.1.89.97.1.1.7.7.100.101.102.97.117.108.116 i 1
```

**1) To return default settings, change the values to Deny(0).****2) Default is converted from ASCII to HEX using a table, which can be found at <https://ru.wikipedia.org/wiki/ASCII>.**

### Enabling/disabling the Framed-IP-Address option (Option 8) in Access-Request packets

**MIB:** eltAAA.mib

**Tables used:** elllRadiusAttrFramedIPAddressAccessEnable — 1.3.6.1.4.1.35265.1.23.80.1.3.0

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.80.1.3.0 i {enable(1), disable(2)}
```

#### **Example of enabling the Framed-IP-Address option (Option 8) in Access-Request packets**

CLI command:

```
radius-server attributes framed-ip-address include-in-access-req
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.80.1.3.0 i 1
```

### Enabling/disabling the NAS-Identifier option (Option 32) in Access-Request packets

**MIB:** eltAAA.mib

**Tables used:** elllRadiusAttrNasIdAccessEnable — 1.3.6.1.4.1.35265.1.23.80.1.1.0

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.80.1.1.0 i {enable(1), disable(2)}
```

#### **Example of enabling the NAS-Identifier option (Option 32) in Access-Request packets**

CLI command:

```
radius-server attributes nas-id include-in-access-req
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.80.1.1.0 i 1
```

### Adding a specified server to a list of used RADIUS servers

**MIB:** rlAAA.mib

**Tables used:** rlRadiusServerInetTable — 1.3.6.1.4.1.89.80.8

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.80.8.1.2.1.4.{ip address (DEC)}.{default UDP port 1812}.{default
UDP port 1813} x "{ip adress(HEX)}" \
1.3.6.1.4.1.89.80.8.1.1.1.4.{ip address (DEC)}.{default UDP port 1812}.{default
UDP port 1813} i {ipv4(1), ipv6(2), ipv4z(3)} \
1.3.6.1.4.1.89.80.8.1.3.1.4.{ip address(DEC)}.{default UDP port 1812}.{default
UDP port 1813} i {default UDP port 1812} \
1.3.6.1.4.1.89.80.8.1.4.1.4.{ip address(DEC)}.{default UDP port 1812}.{default
UDP port 1813} i {default UDP port 1813} \
1.3.6.1.4.1.89.80.8.1.9.1.4.{ip address (DEC)}.{default UDP port 1812}.{default
UDP port 1813} s "#{encoding key}" \
1.3.6.1.4.1.89.80.8.1.13.1.4.{ip address (DEC)}.{default UDP port 1812}.{default
UDP port 1813} i {createAndGo(4), destroy(6)}
```

#### **Example of adding a server to the list of used RADIUS servers**

CLI Command:

```
radius-server host 192.168.1.10 encrypted key da90833f59be
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.80.8.1.2.1.4.192.168.1.10.1812.1813 x "c0a8010a" \
1.3.6.1.4.1.89.80.8.1.1.1.4.192.168.1.10.1812.1813 i 1 \
1.3.6.1.4.1.89.80.8.1.3.1.4.192.168.1.10.1812.1813 i 1812 \
1.3.6.1.4.1.89.80.8.1.4.1.4.192.168.1.10.1812.1813 i 1813 \
1.3.6.1.4.1.89.80.8.1.9.1.4.192.168.1.10.1812.1813 s "#da90833f59be" \
1.3.6.1.4.1.89.80.8.1.13.1.4.192.168.1.10.1812.1813 i 4
```

## 16.6 Loopback detection mechanism

### Global enabling of loopback-detection

**MIB:** rllbd.mib

**Tables used:** rllbdEnable — 1.3.6.1.4.1.89.127.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.127.1.0 i { true(1), false(2) }
```

### **Example of global enabling of loopback-detection**

**CLI Command:**

```
loopback-detection enable
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.127.1.0 i 1
```

### Changing the loopback-detection interval

**MIB:** rllbd.mib

**Tables used:** rllbdDetectionInterval — 1.3.6.1.4.1.89.127.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.127.2.0 I { seconds 1-60 }
```

### **Example of changing loopback frames for 23 seconds**

**CLI command:**

```
loopback-detection interval 23
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.127.2.0 i 23
```

### Changing loopback-detection operation mode

**MIB:** rllbd.mib

**Tables used:** rllbdMode — 1.3.6.1.4.1.89.127.3

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.127.3.0 i {source-mac-addr(1),base-mac-addr(2), multicast-mac-addr(3),broadcast-mac-addr (4) }
```

### Example of changing loopback operation mode to source-mac-addr

CLI command:  

```
loopback-detection mode src-mac-addr
```

SNMP command:  

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.127.3.0 i 1
```

### Enabling/disabling loopback-detection on interfaces

**MIB:** rllbd.mib

**Tables used:** rllbdPortAdminStatus — 1.3.6.1.4.1.89.127.4.1.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.127.4.1.1.{ifindex} i { enable(1), disable(2)}
```

### Example of enabling loopback-detection on TenGigabitEthernet 1/0/2

CLI command:  

```
interface TenGigabitEthernet 1/0/2
loopback-detection enable
```

SNMP command:  

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.127.4.1.1.106 i 1
```

### Viewing loopback-detection operation status on an interface

**MIB:** rllbd.mib

**Tables used:** rllbdPortOperStatus — 1.3.6.1.4.1.89.127.4.1.2

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.127.4.1.2.{ifindex}
```

### Example of viewing loopback-detection state on the GigabitEthernet 1/0/2 interface

CLI command:  

```
show loopback-detection GigabitEthernet 1/0/2
```

SNMP command:  

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.89.127.4.1.2.50
```



#### When using the snmp command:

- 1 — inactive state;
- 2 — active state;
- 3 — loopdetected.

### Viewing blocked VLANs in the vlan-based mode

**MIB:** rllbd.mib

**Tables used:** eltMesLdb — 1.3.6.1.4.1.35265.1.23.127

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.127.4.1.3.{ifindex}.{vlan}
```

#### **Example of viewing vlan 2 state on the GigabitEthernet 1/0/2 interface**

CLI command:

```
show loopback-detection GigabitEthernet 1/0/2
```

SNMP command:

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.127.4.1.3.50.2
```



#### **Possible states:**

- 1 — active**
- 2 — blocked**

## **16.7 Broadcast storm control**

### Configuring storm-control on an interface

**MIB:** radlan-mib.mib

**Tables used:** rlStormCtrl — 1.3.6.1.4.1.89.77

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.77.12.1.2.{ifindex}.{broadcast(1),multicastRegistered(2),multicas
tUnregistered(3), multicastAll(4), unknownUnicast(5)} u {rate} \
1.3.6.1.4.1.89.77.12.1.3.{ifindex}.{broadcast(1),multicastRegistered
(2),multicastUnregistered(3),multicastAll(4), unknownUnicast(5)} I
kiloBitsPerSecond(1),precentaged(2)} \
1.3.6.1.4.1.89.77.12.1.4.{ifindex}.{broadcast(1),multicastRegistered
(2),multicastUnregistered(3), multicastAll(4), unknownUnicast(5)} i
{none(1),trap(2), shutdown(3),trapAndShutdown(4)}
```

#### **Example of enabling storm-control for broadcast traffic on the GigabitEthernet 1/0/1 interface**

CLI command:

```
interface GigabitEthernet 1/0/1
storm-control broadcast kbps 10000 trap shutdown
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.77.12.1.3.49.1 i 1 \
1.3.6.1.4.1.89.77.12.1.2.49.1 u 1000 \
1.3.6.1.4.1.89.77.12.1.4.49.1 i 4
```

### Example of disabling storm-control for broadcast traffic on the GigabitEthernet 1/0/1 interface

**CLI command:**

```
interface GigabitEthernet 1/0/1
 no storm-control broadcast
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.77.12.1.2.49.1 u 0
```

### Enabling/disabling storm-control for unknown unicast-traffic

**MIB:** radlan-stormctrl.mib

**Tables used:** rlStormCtrlRateLimCfgTable — 1.3.6.1.4.1.89.77.12

```
snmpset -v2c -c <community> <IP address> \
iso.3.6.1.4.1.89.77.12.1.2.{ifIndex}.5 u {Kbps,disable (0)}
```

### Example of enabling control of unknown unicast traffic up to 50 kbps

**CLI command:**

```
interface GigabitEthernet 1/0/2
 storm-control unicast Kbps 50
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.77.12.1.2.50.5 u 50
```

## 17 CONFIGURING IP AND MAC ACL

### Creating a mac access-list

**MIB:** qosclimib.mib

**Tables used:** rlQosAcITable — 1.3.6.1.4.1.89.88.7

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.7.1.2.{index-of-acl} s "{name-of-acl}" \
1.3.6.1.4.1.89.88.7.1.3.{index-of-acl} i {type-of-acl: mac(1), ip (2)} \
1.3.6.1.4.1.89.88.7.1.4.{index-of-acl} i {createAndGo(4), destroy(6)}
```

#### **Example of creating MAC ACL with index 207**

CLI command:

```
mac access-list extended 7-mac
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.7.1.2.207 s "7-mac" \
1.3.6.1.4.1.89.88.7.1.3.207 i 1 \
1.3.6.1.4.1.89.88.7.1.4.207 i 4
```

### Creating ip access-list (ACL)

**MIB:** qosclimib.mib

**Tables used:** rlQosAcITable — 1.3.6.1.4.1.89.88.7

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.7.1.2.{index-of-acl} s "{name-of-acl}" \
1.3.6.1.4.1.89.88.7.1.3.{index-of-acl} i {type-of-acl: mac(1), ip (2)} \
1.3.6.1.4.1.89.88.7.1.4.{index-of-acl} i {createAndGo(4), destroy(6)}
```

#### **Example of creating IP ACL with index 107**

CLI command:

```
ip access-list extended 7-ip
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.7.1.2.107 s "7-ip" \
1.3.6.1.4.1.89.88.7.1.3.107 i 2 \
1.3.6.1.4.1.89.88.7.1.4.107 i 4
```



**Example of filling ACL with rules is described in detail in section "Appendix B: Example of creating a standard IP ACL".**

### Binding IP or MAC ACL to a port

**MIB:** qosclimib.mib

**Tables used:**

rlQosIfAcIIn — 1.3.6.1.4.1.89.88.13.1.14

rlQosIfPolicyMapStatus — 1.3.6.1.4.1.89.88.13.1.13



```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.13.1.14.{ifIndex}.2 i {Index-of-acl} \
1.3.6.1.4.1.89.88.13.1.13.{ifIndex}.2 i 1
```

### Example of assigning a rule with index 107 (name ACL 7-ip) to the port GigabitEthernet 1/0/2

CLI command:

```
interface GigabitEthernet 1/0/2
service-acl input 7-ip
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.13.1.14.50.2 i 107 \
1.3.6.1.4.1.89.88.13.1.13.50.2 i 1
```



To remove ACL from the port, the ACL index should be replaced by 0.  
**snmpset -c -v2c private 192.168.1.30 1.3.6.1.4.1.89.88.13.1.14.50.2 i 0**  
**1.3.6.1.4.1.89.88.13.1.13.50.2 i 1**

### Binding IP and MAC ACL to a port

MIB: qosclimib.mib

Tables used:

rIQosIfAclIn — 1.3.6.1.4.1.89.88.13.1.14

rIQosIfIpv6AclIn — 1.3.6.1.4.1.89.88.13.1.201.3.6.1.4.1.89.88.13.1.20

rIQosIfPolicyMapStatus — 1.3.6.1.4.1.89.88.13.1.13

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.13.1.14.{Ifindex}.2 i {Index-of-mac-acl} \
1.3.6.1.4.1.89.88.13.1.20.{Ifindex}.2 i {Index-of-ip-acl} \
1.3.6.1.4.1.89.88.13.1.13.{ifIndex}.2 i 1
```

### Example of assigning a rule with an index of 107 and 207 (name ACL 7-ip for IP ACL and 7-mac for MAC ACL) to GigabitEthernet port 1/0/2 (Ifindex 50).

CLI command:

```
interface GigabitEthernet 1/0/2
service-acl input 7-mac 7-ip
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.13.1.14.50.2 i 207 \
1.3.6.1.4.1.89.88.13.1.20.50.2 i 107 \
1.3.6.1.4.1.89.88.13.1.13.50.2 i 1
```



To remove ACL from the port, the IP index and MAC ACL should be replaced by 0.

**snmpset -v2c -c private 192.168.1.30 \**  
**1.3.6.1.4.1.89.88.13.1.14.50.2 i 0 \**  
**1.3.6.1.4.1.89.88.13.1.20.50.2 i 0 \**  
**1.3.6.1.4.1.89.88.13.1.13.50.2 i 1**

## Creating a policy-map and binding an ACL to it

**MIB:** qosclimib.mib

### Tables used:

rIQosClassMapTable — 1.3.6.1.4.1.89.88.9

rIQosPolicyMapTable — 1.3.6.1.4.1.89.88.11

rIQosPolicyClassPriorityRefTable — 1.3.6.1.4.1.89.88.39

**Scheme:** the creation of a policy-map is done in several queries

#### 1. Creating a class and assigning properties

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.9.1.2.{index-of-class} s "{name-of-class-map}" \
1.3.6.1.4.1.89.88.9.1.3.{index-of-class} i {matchAll (1)} \
1.3.6.1.4.1.89.88.9.1.7.{index-of-class} i {index-of-acl} \
1.3.6.1.4.1.89.88.9.1.9.{index-of-class} i {Mark vlan disable (1), enable(2)} \
1.3.6.1.4.1.89.88.9.1.13.{index-of-class} i {create and go(4),destroy(6)}
```

#### 2. Creating a policy-map and enabling it

```
snmpset -v2c -with <community> <IP address> \
1.3.6.1.4.1.89.88.11.1.2.{index-of-policy-map} s {name-of-policy-map} \
1.3.6.1.4.1.89.88.11.1.3.{index-of-policy-map} i {createAndGo(4), destroy(6)}
```

#### 3. Binding class-map to policy-map

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.39.1.2.1.{index-of-class} i {index-of-class} \
1.3.6.1.4.1.89.88.39.1.3.1.{index-of-class} i {index-of-policy-map}
```

#### 4. Creating a speed limit for class-map

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.10.1.2.{Number-of-class-in-policy} s {Policeman-cm-20} \
1.3.6.1.4.1.89.88.10.1.3.{Number-of-class-in-policy} i {single(1), aggregate(2)} \
1.3.6.1.4.1.89.88.10.1.4.{Number-of-class-in-policy} i {rate} \
1.3.6.1.4.1.89.88.10.1.5.{Number-of-class-in-policy} i {burst} \
1.3.6.1.4.1.89.88.10.1.6.{Number-of-class-in-policy} i {none(1), drop(2), remark(3)} \
1.3.6.1.4.1.89.88.10.1.8.{Number-of-class-in-policy} i {createAndGo(4), destroy(6)}
```

#### 5. Binding the speed limit to class-map

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.9.1.6.{index-of-class} i {Number-of-class-in-policy}
```

#### 6. Setting the label value to DSCP and/or cos traffic, specifying the output queue

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.88.5.1.1.{index-of-class}.setDSCP(3), setQueue(4), setCos(5)} i {setDSCP(3),
setQueue(4), setCos(5)} \
1.3.6.1.4.1.35265.1.23.88.5.1.2.{index-of-class}.setDSCP(3), setQueue(4), setCos(5)} i {Mark value of
DSCP/queue/cos(DEC)} \
1.3.6.1.4.1.35265.1.23.88.5.1.3.{index-of-class}.setDSCP(3), setQueue(4), setCos(5)} i {createAndGo(4),
destroy(6)}
```

**Example of binding an IP ACL with index-of-acl = 107 to a class-map named test and labeling DSCP = 36(DEC), cos = 4 and queue = 8 for traffic covered by IP ACL. Class test is binded to a policy-map with test1 name**

CLI command:

```
qos advanced
 ip access-list extended 7-ip
  permit ip any any any any
exit

class-map test
 match access-group 7-ip
exit
policy-map test1
 class test
  set dscp 36
  set queue 8
  set cos 4
  police 97000 524288 exceed-action drop
exit
exit
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.9.1.2.20 s "test" \
1.3.6.1.4.1.89.88.9.1.3.20 i 1 \
1.3.6.1.4.1.89.88.9.1.7.20 i 107 \
1.3.6.1.4.1.89.88.9.1.9.20 i 1 \
1.3.6.1.4.1.89.88.9.1.13.20 i 4

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.11.1.2.1 s "test1" \
1.3.6.1.4.1.89.88.11.1.3.1 i 4

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.39.1.2.1.20 i 20 \
1.3.6.1.4.1.89.88.39.1.3.1.20 i 1

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.10.1.2.1 s "Policer-cm-20" \
1.3.6.1.4.1.89.88.10.1.3.1 i 1 \
1.3.6.1.4.1.89.88.10.1.4.1 u 97000 \
1.3.6.1.4.1.89.88.10.1.5.1 u 524288 \
1.3.6.1.4.1.89.88.10.1.6.1 i 2 \
1.3.6.1.4.1.89.88.10.1.8.1 i 4

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.9.1.6.20 i 1

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.88.5.1.1.20.3 i 3 \
1.3.6.1.4.1.35265.1.23.88.5.1.2.20.3 i 36 \
1.3.6.1.4.1.35265.1.23.88.5.1.3.20.3 i 4

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.88.5.1.1.20.4 i 4 \
1.3.6.1.4.1.35265.1.23.88.5.1.2.20.4 i 8 \
1.3.6.1.4.1.35265.1.23.88.5.1.3.20.4 i 4

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.88.5.1.1.20.5 i 5 \
1.3.6.1.4.1.35265.1.23.88.5.1.2.20.5 i 4 \
1.3.6.1.4.1.35265.1.23.88.5.1.3.20.5 i 4
```

### Assigning Policy-map to a port

**MIB:** qosclimib.mib

**Tables used:** rIQosIfPolicyMapPointerIn — 1.3.6.1.4.1.89.88.13.1.3

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.88.13.1.3.{Ifindex}.2 i {Index-of-policy-map}
```

#### **Example of assigning a policy-map with index 1 to the port GigabitEthernet 1/0/3**

CLI command:

```
interface GigabitEthernet 1/0/3  
service-policy input test1
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.88.13.1.3.51.2 i 1
```

## 18 CONFIGURING PROTECTION AGAINST DOS ATTACKS

### Enabling security-suite

**MIB:** rISecuritySuiteMib

**Tables used:**

rISecuritySuiteGlobalEnable — 1.3.6.1.4.1.89.120.1

```
snmpset -v2c -c <community> <IP address> 1.3.6.1.4.1.89.120.1.0 i {enable-
global-rules-only (1), enable- all-rules-types (2), disable (3)}
```

#### **Example of enabling security-suite command class for all rules**

CLI command:

```
security-suite enable
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.120.1.0 i 2
```

### Configuring security-suite operation mode

**MIB:** rISecuritySuiteMib

**Tables used:** rISecuritySuiteSynProtectionMode — 1.3.6.1.4.1.89.120.10

```
snmpset -v2c -c <community> <IP address> 1.3.6.1.4.1.89.120.10.0 i {disabled
(1), report (2), block (3)}
```

#### **Example of enabling "report" operation mode**

CLI command:

```
security-suite syn protection mode report
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.120.10.0 i 2
```

### Switch off protection against tcp packets with simultaneously set SYN and FIN flags

**MIB:** rISecuritySuiteMib

**Tables used:** rISecuritySuiteDenySynFinTcp — 1.3.6.1.4.1.89.120.9

```
snmpset -v2c -c <community> <IP address> 1.3.6.1.4.1.89.120.9.0 i {(deny (1),
permit (2)}
```

#### **Example of disabling protection against TCP packets with SYN and FIN flags set**

CLI command:

```
security-suite deny syn-fin
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
```

## 19 QUALITY OF SERVICE (QOS)

### 19.1 QoS configuration

#### Limiting uplink bandwidth on Ethernet ports

**MIB:** qosclimib.mib

**Tables used:** rIQosIfPolicyEntry — 1.3.6.1.4.1.89.88.13.1

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.88.13.1.6.{port ifindex}.2 i {disable(1),enable
(1)} \
  1.3.6.1.4.1.89.88.13.1.7.{port ifindex}.2 i {traffic-shape} \
  1.3.6.1.4.1.89.88.13.1.8.{port ifindex}.2 i {Burst size in bytes}
```

#### **Example of limiting the uplink bandwidth on the interface to a value of 20 Mbps**

CLI command:

```
interface GigabitEthernet 1/0/1
traffic-shape 20480 500000
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.88.13.1.6.49.2 i 2 \
  1.3.6.1.4.1.89.88.13.1.7.49.2 i 20480 \
  1.3.6.1.4.1.89.88.13.1.8.49.2 i 500000
```

#### Limiting downlink bandwidth on Ethernet ports

**MIB:** radlan-mib.mib

**Tables used:** rIStormCtrlRateLimCfgTable — 1.3.6.1.4.1.89.77.12

```
snmpset -v2c -c <community> <IP address> \
  1.3.6.1.4.1.89.77.12.1.2.{ifIndex}.6 u {limit} \
  1.3.6.1.4.1.89.77.12.1.5.{ifIndex}.6 u {Burst size in bytes}
```

#### **Example of limiting the downlink bandwidth on the interface GigabitEthernet 1/0/1 to a value of 10 Mbps**

CLI command:

```
interface GigabitEthernet 1/0/1
rate-limit 10240 burst 500000
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
  1.3.6.1.4.1.89.77.12.1.2.49.6 u 10240 \
  1.3.6.1.4.1.89.77.12.1.5.49.6 u 500000
```



**To disable rate-limit on an interface, the following must be done (on the example of the Gigabitethernet 1/0/1 interface):**

```
snmpset -v2c -c private 192.168.1.30 1.3.6.1.4.1.89.77.12.1.2.49.6 u 0
1.3.6.1.4.1.89.77.12.1.5.49.6 u 128000
```

### Creating a qos tail-drop profile and expanding queue descriptors

**MIB:** eltQosTailDropMIB.mib

**Tables used:** eltQosTailDropProfileQueueTable — 1.3.6.1.4.1.35265.1.23.12.1.1.1

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.12.1.1.1.1.4.{Profile number (1-4)}.{Queue number(1-8)}
i {size (0-400)}
```

#### **Example of creating a qos tail-drop qos profile and expanding descriptors for queues**

CLI command:

```
qos tail-drop profile 2
queue 1 limit 400
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.12.1.1.1.1.4.2.1 i 400
```



**To return to the default settings, set the value to 12.**

### Setting the size of packet pool for the port

**MIB:** eltQosTailDropMIB.mib

**Tables used:** eltQosTailDropProfileTable — 1.3.6.1.4.1.35265.1.23.12.1.1.4

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.12.1.1.4.1.2{profile number(1-4)} i {size (0-400)}
```

#### **Example of setting the size of a packet pool**

CLI command:

```
qos tail-drop profile 2
port-limit 400
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.12.1.1.4.1.2.2 i 400
```

### Assigning a created profile to an interface

**MIB:** eltQosTailDropMIB.mib

**Tables used:** eltQosTailDropIfConfigTable — 1.3.6.1.4.1.35265.1.23.12.1.1.2

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.12.1.1.2.1.1.{IfIndex} i {profile number (1-4)}
```

#### **Example of assigning a profile to the GigabitEthernet 1/0/1 interface**

CLI command:

```
interface GigabitEthernet 1/0/1
qos tail-drop profile 2
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.12.1.1.2.1.1.49 i 2
```

**View display of global limits, descriptors, buffers****MIB:** radlan-mib.mib**Tables used:** eltQosTailDropConfigTable — 1.3.6.1.4.1.35265.1.23.12.1.1.3

```
snmpwalk -v2c -c <community> <ip address> \  
1.3.6.1.4.1.35265.1.23.12.1.1.3
```

**Example of viewing the display of global limits, descriptors, buffers****CLI command:**

```
show qos tail-drop
```

**SNMP command:**

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.12.1.1.3
```

**View output table of current allocated qos resources (limits, descriptors, buffers)****MIB:** ELTEX-MES-QOS-TAIL-DROP-MIB**Tables used:** eltQosTailDropStatusTable — 1.3.6.1.4.1.35265.1.23.12.1.2.1

```
snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.1.23.12.1.2.1
```

**Example of viewing the output of current limits, descriptors, buffers****CLI command:**

```
show qos tail-drop
```

**SNMP command:**

```
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.12.1.2.1
```

**Remarking DSCP to CoS****MIB:** eltQosclimib.mib**Tables used:** eltQosCos — 1.3.6.1.4.1.35265.1.23.88.6.1.2

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.35265.1.23.88.6.1.2.{DSCP mark} i {CoS mark}
```

**Example of DSCP 30 remarking to mark 5 COS****CLI command:**

```
qos map dscp-cos 30 to 5
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.35265.1.23.88.6.1.2.30 i 5
```



### Limiting multicast packet replication

**MIB:** eltQoSTailDropMIB.mib

**Tables used:** eltQoSTailDropMulticastReplicationLimitEnable — 1.3.6.1.4.1.35265.1.23.12.1.1.8

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.12.1.1.8.0 i {enabled (1), disabled (2)}
```

#### **Example of enabling multicast packet replication restrictions**

CLI command:  

```
qos tail-drop multicast replication-limit
```

SNMP command:  

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.12.1.1.8.0 i 1
```

## 19.2 QoS statistics

### Viewing Tail Drop counters per queue

**MIB:** eltMesCounters.mib

**Tables used:** eltMesCountersMIB — 1.3.6.1.4.1.35265.1.23.1.8

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.1.8.1.2.1.1.1.{Dropped packets(5), Passed
packets(7)}.{ifIndex}.{1-8}.0
```

#### **Example of viewing counters on the first queue**

CLI command:  

```
show interface GigabitEthernet1/0/6
```

SNMP command:  

```
snmpwalk -v2c -c public 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.1.8.1.2.1.1.1.7.54.1.0
```

## 20 ROUTING

### 20.1 Static routing

#### Viewing routing table

**MIB:** radlan-mib.mib

**Tables used:** ipCidrRouteTable — 1.3.6.1.2.1.4.24.4

```
snmpwalk -v2c -c <community> <IP address> \
1.3.6.1.2.1.4.24.4
```

### Example of viewing the routing table

CLI command:  
show ip route

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.2.1.4.24.4

#### Viewing static routes

**MIB:** rlip.mib

**Tables used:** rIIPStaticRouteTable — 1.3.6.1.4.1.89.26.17.1

snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.26.17.1

### Example of viewing static routes

CLI command:  
show running-config routing

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.26.17.1

## 20.2 Dynamic routing

#### Viewing OSPF neighbourhood

**MIB:** rlip.mib

**Tables used:** rIOSpfNbrTable — 1.3.6.1.4.1.89.210.11

snmpwalk -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.210.11

### Example of viewing an OSPF neighborhood

CLI command:  
show ip ospf neighbor

SNMP command:  
snmpwalk -v2c -c public 192.168.1.30 \  
1.3.6.1.4.1.89.210.11



## APPENDIX B: EXAMPLE OF CREATING A STANDARD IP ACL

This annex describes an example of filling an IP ACL with index-of-acl = 107 with the following rules:

```
ip access-list extended 7-ip
 deny udp any bootps any bootpc ace-priority 20
 permit igmp any any ace-priority 40
 deny ip any any any 224.0.0.0 15.255.255.255 ace-priority 60
 permit ip any any 37.193.119.7 0.0.0.0 any ace-priority 80
 permit ip any any 10.130.8.3 0.0.0.0 any ace-priority 100
 permit ip any any 192.168.0.0 0.0.0.15 any ace-priority 120
 permit ip 00:19:16:15:14:16 00:00:00:00:00:00 any 37.193.119.7 0.0.0.0 any
 ace-priority 140
 permit ip any 01:00:0c:00:00:00 00:00:00:ff:ff:ff any any ace-priority 160
 exit
```

### Creating a deny udp any bootps any bootpc rule

**MIB:** qosclimib.mib

**Tables used:** rIQosTupleTable — 1.3.6.1.4.1.89.88.5, rIQosAceTidxTable — 1.3.6.1.4.1.89.88.31

**Scheme:** the rule is created in two requests.

#### 1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
 1.3.6.1.4.1.89.88.5.1.2.{value of field 1} i {protocol(1)} \
 1.3.6.1.4.1.89.88.5.1.4.{value of field 1} x {protocol index (HEX)} \
 1.3.6.1.4.1.89.88.5.1.3.{value of field 1} i {Value in port table for protocol
 = 0. Constant for this rule} \
 1.3.6.1.4.1.89.88.5.1.2.{value of field 2} i {udp-port-src(6)} \
 1.3.6.1.4.1.89.88.5.1.3.{value of field 2} i {Number of source port (DEC)} \
 1.3.6.1.4.1.89.88.5.1.4.{value of field 2} x {source ip(HEX)} \
 1.3.6.1.4.1.89.88.5.1.2.{value of field 3} i { udp-port-dst(6)} \
 1.3.6.1.4.1.89.88.5.1.3.{value of field 3} i {Number of dst port (DEC)} \
 1.3.6.1.4.1.89.88.5.1.4.{value of field 3} x {dst ip(HEX)}
```

#### 2. Binding a rule by index-of-rule to an ACL by index-of-acl as deny.

```
snmpset -v2c -c <community> <IP address> \
 1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {deny(2)} \
 1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {udp(3)} \
 1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 1} \
 1.3.6.1.4.1.89.88.31.1.7.{index-of-acl}.{index-of-rule} i {value of field 3} \
 1.3.6.1.4.1.89.88.31.1.9.{index-of-acl}.{index-of-rule} i {value of field 2}
```

**Example of adding a deny udp any bootpc rule to IP ACL 7-ip (since the rule is assumed to be the first one, then index-of-rule=20)**

CLI command:

```
ip access-list extended 7-ip
 deny udp any bootps any bootpc ace-priority 20
 exit
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
 1.3.6.1.4.1.89.88.5.1.2.1 i 1 \
 1.3.6.1.4.1.89.88.5.1.4.1 x "0x11 FF" \
```

```

1.3.6.1.4.1.89.88.5.1.3.1 i 0 \
1.3.6.1.4.1.89.88.5.1.2.2 i 6 \
1.3.6.1.4.1.89.88.5.1.3.2 i 67 \
1.3.6.1.4.1.89.88.5.1.4.2 x "0x00 00" \
1.3.6.1.4.1.89.88.5.1.2.3 i 7 \
1.3.6.1.4.1.89.88.5.1.3.3 i 68 \
1.3.6.1.4.1.89.88.5.1.4.3 x "0x00 00"

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.31.1.3.107.20 i 2 \
1.3.6.1.4.1.89.88.31.1.4.107.20 i 3 \
1.3.6.1.4.1.89.88.31.1.5.107.20 i 1 \
1.3.6.1.4.1.89.88.31.1.7.107.20 i 2 \
1.3.6.1.4.1.89.88.31.1.9.107.20 i 3

```

### Creating a permit igmp any any rule

**MIB:** qosclimib.mib

#### **Tables used:**

rlQoSTupleTable — 1.3.6.1.4.1.89.88.5

rlQoSAceTidxTable — 1.3.6.1.4.1.89.88.31

**Scheme:** a rule is created in two requests.

#### 1. Setting the rule parameters.

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.5.1.2.{value of field 4} i {protocol(1)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 4} x {protocol index (HEX)}

```

#### 2. Binding a rule by index-of-rule to an ACL by index-of-acl as permit.

```

snmpset -v2c -c <community> <IP address> \

```

```

1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {permit (1)} \
1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {igmp (8)} \
1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 4}

```

**Example of adding a permit igmp any rule to IP ACL 7-ip (since the rule is assumed to be the second one, the index-of-rule=40)**

#### CLI command:

```

ip access-list extended 7-ip
 permit igmp any any ace-priority 40
exit

```

#### SNMP command:

```

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.5.1.2.4 i 1 \
1.3.6.1.4.1.89.88.5.1.4.4 x "0x02 FF"

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.31.1.3.107.40 i 1 \
1.3.6.1.4.1.89.88.31.1.4.107.40 i 8 \
1.3.6.1.4.1.89.88.31.1.5.107.40 i 4

```

### Creating a deny ip any any any 224.0.0.0 15.255.255.255 rule

**MIB:** qosclimib.mib

**Tables used:**

rIQosTupleTable — 1.3.6.1.4.1.89.88.5

rIQosAceTidxTable — 1.3.6.1.4.1.89.88.31

**Scheme:** a rule is created in two requests.

1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.5.1.2.{value of field 5} i {ip-dest(3)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 5} x {dst ip +wildcard mask (HEX)}
```

2. Binding a rule by index-of-rule to an ACL by index-of-acl as deny.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {deny (2)} \
1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {ip (1)} \
1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 5}
```

**Example of adding a deny ip any any any 224.0.0.0 15.255.255.255 rule to IP ACL 7-ip (since the rule is assumed to be the third one, then index-of-rule=60)**

CLI command:

```
ip access-list extended 7-ip
deny ip any any any 224.0.0.0 15.255.255.255 ace-priority 60
exit
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.5.1.2.5 i 3 \
1.3.6.1.4.1.89.88.5.1.4.5 x "0xE0 00 00 00 0F FF FF FF"
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.31.1.3.107.60 i 2 \
1.3.6.1.4.1.89.88.31.1.4.107.60 i 1 \
1.3.6.1.4.1.89.88.31.1.5.107.60 i 5
```

### Creating a permit ip any any 37.193.119.7 0.0.0.0 any rule

**MIB:** qosclimib.mib

**Tables used:**

rIQosTupleTable — 1.3.6.1.4.1.89.88.5

rIQosAceTidxTable — 1.3.6.1.4.1.89.88.31

**Scheme:** a rule is created in two requests.

1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.5.1.2.{value of field 6} i {ip-source(2)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 6} x {source ip +wildcard mask (HEX)}
```

## 2. Binding a rule by index-of-rule to an ACL by index-of-acl as permit.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {permit (1)} \
1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {ip (1)} \
1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 6}
```

**Example of adding a permit ip any any 37.193.119.7 0.0.0.0 any to IP ACL 7-ip (since the rule is assumed to be the fourth one, the index-of-rule=80)**

CLI command:

```
ip access-list extended 7-ip
 permit ip any any 37.193.119.7 0.0.0.0 any ace-priority 80
exit
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.5.1.2.6 i 2 \
1.3.6.1.4.1.89.88.5.1.4.6 x "0x25 C1 77 07 00 00 00 00"
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.31.1.3.107.80 i 1 \
1.3.6.1.4.1.89.88.31.1.4.107.80 i 1 \
1.3.6.1.4.1.89.88.31.1.6.107.80 i 6
```

### Creating a permit ip any any 10.130.8.3 0.0.0.0 any rule

MIB: qosclimib.mib

Tables used:

rIQoSupleTable — 1.3.6.1.4.1.89.88.5

rIQoSAceTidxTable — 1.3.6.1.4.1.89.88.31

**Scheme:** a rule is created in two requests.

#### 1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.5.1.2.{value of field 7} i {ip-source(2)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 7} x {source ip +wildcard mask (HEX)}
```

#### 2. Binding a rule by index-of-rule to an ACL by index-of-acl as permit

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {permit (1)} \
1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {ip (1)} \
1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 7}
```

**Example of adding a permit ip any any 10.130.8.3 0.0.0.0 any to IP ACL 7-ip (since the rule is assumed to be the fifth one, the index-of-rule=100)**

CLI command:

```
ip access-list extended 7-ip
 permit ip any any 10.130.8.3 0.0.0.0 any ace-priority 100
exit
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.5.1.2.7 i 2 \
1.3.6.1.4.1.89.88.5.1.4.7 x "0x0A 82 08 03 00 00 00 00"
```

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.31.1.3.107.100 i 1 \
1.3.6.1.4.1.89.88.31.1.4.107.100 i 1 \
1.3.6.1.4.1.89.88.31.1.6.107.100 i 7
```

**Creating a permit ip any any 192.168.0.0 0.0.0.15 any rule**

**MIB:** qosclimib.mib

**Tables used:**

rIQoSTupleTable — 1.3.6.1.4.1.89.88.5  
rIQoSAceTidxTable — 1.3.6.1.4.1.89.88.31

**Scheme:** a rule is created in two requests.

1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.5.1.2.{value of field 8} i {ip-source(2)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 8} x {source ip +wildcard mask (HEX)}
```

2. Binding a rule by index-of-rule to an ACL by index-of-acl as permit.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {permit (1)} \
1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {ip (1)} \
1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 8}
```

**Example of adding a permit ip any any 192.168.0.0 0.0.0.15 any to IP ACL 7-ip (since the rule is assumed to be the sixth one, the index-of-rule=120)**

**CLI command:**

```
ip access-list extended 7-ip
 permit ip any any 192.168.0.0 0.0.0.15 any ace-priority 120
exit
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.5.1.2.8 i 2 \
1.3.6.1.4.1.89.88.5.1.4.8 x "0xC0 A8 00 00 00 00 00 0F"

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.31.1.3.107.120 i 1 \
1.3.6.1.4.1.89.88.31.1.4.107.120 i 1 \
1.3.6.1.4.1.89.88.31.1.6.107.120 i 8
```

**Creating a permit ip 00:19:16:15:14:16 00:00:00:00:00:00 any 37.193.119.7 0.0.0.0 any rule**

**MIB:** qosclimib.mib

**Tables used:**

rIQoSTupleTable — 1.3.6.1.4.1.89.88.5  
rIQoSAceTidxTable — 1.3.6.1.4.1.89.88.31



**Scheme:** a rule is created in two requests.

1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.5.1.2.{value of field 9} i {ip-source(2)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 9} x {source ip +wildcard mask (HEX)} \
1.3.6.1.4.1.89.88.5.1.2.{value of field 10} i {mac-src(10)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 10} x {source mac +wildcard mask (HEX)}
```

2. Binding a rule by index-of-rule to an ACL by index-of-acl as permit.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {permit (1)} \
1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {ip (1)} \
1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 9} \
1.3.6.1.4.1.89.88.31.1.6.{index-of-acl}.{index-of-rule} i {value of field 10}
```

**Example of adding a permit ip 00:19:16:15:14:16 00:00:00:00:00:00 any 37.193.119.7 0.0.0.0 any to IP ACL 7-ip (since the rule is assumed to be the seventh one, the index-of-rule=140)**

CLI command:

```
ip access-list extended 7-ip
 permit ip 00:19:16:15:14:16 00:00:00:00:00:00 any 37.193.119.7 0.0.0.0 any ace-
priority 140
exit
```

SNMP command:

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.5.1.2.9 i 2 \
1.3.6.1.4.1.89.88.5.1.4.9 x "0x25 C1 77 07 00 00 00 00" \
1.3.6.1.4.1.89.88.5.1.2.10 i 10 \
1.3.6.1.4.1.89.88.5.1.4.10 x "0x001916151416000000000000"

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.31.1.3.107.140 i 1 \
1.3.6.1.4.1.89.88.31.1.4.107.140 i 1 \
1.3.6.1.4.1.89.88.31.1.5.107.140 i 9 \
1.3.6.1.4.1.89.88.31.1.6.107.140 i 10
```

*Creating a permit ip any 01:00:0c:00:00:00 00:00:00:ff:ff:ff any any rule*

**MIB:** qosclimib.mib

**Tables used:**

rIQosTupleTable — 1.3.6.1.4.1.89.88.5  
rIQosAceTidxTable — 1.3.6.1.4.1.89.88.31

**Scheme:** a rule is created in two requests.

1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
.{value of field 11} i {mac-dest (11)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 11} x {dst mac +wildcard mask (HEX)}
```

## 2. Binding a rule by index-of-rule to an ACL by index-of-acl as permit.

```
snmpset -v2c -c <community> <IP address> \  
1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {permit (1)} \  
1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {ip (1)} \  
1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 11}
```

**Example of adding a permit ip any 01:00:0c:00:00:00 00:00:00:ff:ff:ff any any to IP ACL 7-ip (since the rule is assumed to be the eighth one, the index-of-rule=160)**

**CLI command:**

```
ip access-list extended 7-ip  
 permit ip any 01:00:0c:00:00:00 00:00:00:ff:ff:ff any any ace-priority 160  
exit
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.88.5.1.2.11 i 11 \  
1.3.6.1.4.1.89.88.5.1.4.11 x "0x01000c00000000000000ffffff"  
  
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.88.31.1.3.107.160 i 1 \  
1.3.6.1.4.1.89.88.31.1.4.107.160 i 1 \  
1.3.6.1.4.1.89.88.31.1.5.107.160 i 11
```

## APPENDIX C. EXAMPLE OF CREATING, FILLING AND REMOVING AN OFFSET-LIST WITH MAC ACL

This annex describes an example of creating and filling a MAC ACL with index-of-acl = 207 with the following rules:

```
mac access-list extended 7-mac
offset-list PADO 12 12 00 88 12 13 00 63 12 15 00 07
deny any any offset-list PADO ace-priority 20
```

### Creating a mac access-list

**MIB:** qosclimib.mib

**Tables used:** rIQosAcITable — 1.3.6.1.4.1.89.88.7

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.7.1.2.{index-of-acl} s "{name-of-acl}" \
1.3.6.1.4.1.89.88.7.1.3.{index-of-acl} i {type-of-acl: mac(1), ip (2)} \
1.3.6.1.4.1.89.88.7.1.4.{index-of-acl} i {createAndGo(4), destroy(6)}
```

### **Example of creating MAC ACL with index 207**

**CLI command:**

```
mac access-list extended 7-mac
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.7.1.2.207 s "7-mac" \
1.3.6.1.4.1.89.88.7.1.3.207 i 1 \
1.3.6.1.4.1.89.88.7.1.4.207 i 4
```

### Creating an offset-list

**MIB:** qosclimib.mib

**Tables used:**

rIQosOffsetTable — 1.3.6.1.4.1.89.88.4

eltMesQosCliMib — 1.3.6.1.4.1.35265.1.23.88

### **Example of creating an offset-list PADO I2 12 00 88 I2 13 00 63 I2 15 00 07:**

The rule is created in two requests.

#### 1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.4.1.2.{value of field 1 in offset-list} i {layer2-start(2) -
specifying the packet header or individual header parameters} \
1.3.6.1.4.1.89.88.4.1.3.{value of field 1 in offset-list} i {Byte number in the
header} \
1.3.6.1.4.1.89.88.4.1.4.{value of field 1 in offset-list} i {WildcardMask in
byte in DEC} \
1.3.6.1.4.1.89.88.4.1.5.{value of field 1 in offset-list} i {Byte value
considering WildcardMask in DEC} \
```

```

1.3.6.1.4.1.89.88.4.1.7.{value of field 1 in offset-list} i {createAndGo(4),
destroy(6)}
1.3.6.1.4.1.89.88.4.1.2.{value of field 2 in offset-list} i {layer2-start(2) -
specifying the packet header or individual header parameters} \
1.3.6.1.4.1.89.88.4.1.3.{value of field 2 in offset-list} i {Byte number in the
header} \
1.3.6.1.4.1.89.88.4.1.4.{value of field 2 in offset-list} i {WildcardMask in
byte in DEC} \
1.3.6.1.4.1.89.88.4.1.5.{value of field 2 in offset-list} i {Byte value with
WildcardMask in DEC} \
1.3.6.1.4.1.89.88.4.1.7.{value of field 2 in offset-list} i {createAndGo(4),
destroy(6)}

1.3.6.1.4.1.89.88.4.1.2.{value of field 3 in offset-list} i {layer2-start(2) -
specifying the packet header or individual header parameters} \
1.3.6.1.4.1.89.88.4.1.3.{value of field 3 in offset-list} i {Byte number in the
header} \
1.3.6.1.4.1.89.88.4.1.4.{value of field 3 in offset-list} i {WildcardMask in
byte in DEC} \
1.3.6.1.4.1.89.88.4.1.5.{value of field 3 in offset-list} i {Byte value with
WildcardMask in DEC} \
1.3.6.1.4.1.89.88.4.1.7.{value of field 3 in offset-list} i {createAndGo(4),
destroy(6)}

```

## 2. Binding offset-list to index-of-acl.

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.35265.1.23.88.1.1.1.{index-of-acl}.{Number of letters in the name
of the offset-list}.{Name of the offset-list in DEC, each letter of the name is
separated from the next by a point} i {index-of-acl}
1.3.6.1.4.1.35265.1.23.88.1.1.3.{index-of-acl}.{Number of letters in the name
of the offset-list}.{Name of the offset-list in DEC, each letter of the name is
separated from the next by a point} i {value of field 1 in offset-list} \
1.3.6.1.4.1.35265.1.23.88.1.1.4.{index-of-acl}.{Number of letters in the name
of the offset-list}.{Name of the offset-list in DEC, each letter of the name is
separated from the next by a point} i {value of field 2 in offset-list} \
1.3.6.1.4.1.35265.1.23.88.1.1.5.{index-of-acl}.{Number of letters in the name
of the offset-list}.{Name of the offset-list in DEC, each letter of the name is
separated from the next by a point} i {value of field 3 in offset-list} \
1.3.6.1.4.1.35265.1.23.88.1.1.8.{index-of-acl}.{Number of letters in the name
of the offset-list}.{Name of the offset-list in DEC, each letter of the name is
separated from the next by a point} i {createAndGo(4), destroy(6)}

```

### Example of adding a deny udp any bootps any bootpc rule to MAC ACL 7-mac (since the rule is assumed to be the first one, then index-of-rule=20)

#### CLI command:

```

mac access-list extended 7-mac
offset-list PADO 12 12 00 88 12 13 00 63 12 15 00 07
exit

```

#### SNMP command:

```

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.4.1.2.1 i 2 \
1.3.6.1.4.1.89.88.4.1.3.1 i 12 \
1.3.6.1.4.1.89.88.4.1.4.1 i 0 \
1.3.6.1.4.1.89.88.4.1.5.1 i 136 \
1.3.6.1.4.1.89.88.4.1.7.1 i 4 \
1.3.6.1.4.1.89.88.4.1.2.2 i 2 \
1.3.6.1.4.1.89.88.4.1.3.2 i 13 \
1.3.6.1.4.1.89.88.4.1.4.2 i 0 \
1.3.6.1.4.1.89.88.4.1.5.2 i 99 \
1.3.6.1.4.1.89.88.4.1.7.2 i 4 \

```

```

1.3.6.1.4.1.89.88.4.1.2.3 i 2 \
1.3.6.1.4.1.89.88.4.1.3.3 i 15 \
1.3.6.1.4.1.89.88.4.1.4.3 i 0 \
1.3.6.1.4.1.89.88.4.1.5.3 i 7 \
1.3.6.1.4.1.89.88.4.1.7.3 i 4

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.35265.1.23.88.1.1.1.207.4.80.65.68.79 i 207 \
1.3.6.1.4.1.35265.1.23.88.1.1.3.207.4.80.65.68.79 i 1 \
1.3.6.1.4.1.35265.1.23.88.1.1.4.207.4.80.65.68.79 i 2 \
1.3.6.1.4.1.35265.1.23.88.1.1.5.207.4.80.65.68.79 i 3 \
1.3.6.1.4.1.35265.1.23.88.1.1.8.207.4.80.65.68.79 i 4

```



**Offset-list name is converted from ASCII to HEX using a table, which can be found at <https://ru.wikipedia.org/wiki/ASCII>.**

### Creating a deny any any offset-list PADO rule

**MIB:** qosclimib.mib

#### **Tables used:**

rIQoStupleTable — 1.3.6.1.4.1.89.88.5

rIQoSaceTidxTable — 1.3.6.1.4.1.89.88.31

The rule is created in two requests:

#### 1. Setting the rule parameters.

```

snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.5.1.2.{value of field 1 in ACL} i {general(15)} \
1.3.6.1.4.1.89.88.5.1.2.{value of field 2 in ACL} i {general(15)} \
1.3.6.1.4.1.89.88.5.1.2.{value of field 2 in ACL} i {general(15)} \
1.3.6.1.4.1.89.88.5.1.3.{value of field 1 in ACL} i {value of field 1 in
offset-list} \
1.3.6.1.4.1.89.88.5.1.3.{value of field 2 in ACL} i {value of field 2 in
offset-list} \
1.3.6.1.4.1.89.88.5.1.3.{value of field 3 in ACL} i {value of field 3 in
offset-list} \
1.3.6.1.4.1.89.88.5.1.5.{value of field 1 in ACL} i {createAndGo(4),
destroy(6)} \
1.3.6.1.4.1.89.88.5.1.5.{value of field 2 in ACL} i {createAndGo(4),
destroy(6)} \
1.3.6.1.4.1.89.88.5.1.5.{value of field 3 in ACL} i {createAndGo(4),
destroy(6)}
2. Binding a rule by index-of-rule to an ACL by index-of-acl as deny
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {deny(2)} \
1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {mac(5)} \
1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 1 in
ACL} \
1.3.6.1.4.1.89.88.31.1.6.{index-of-acl}.{index-of-rule} i {value of field 2 in
ACL} \
1.3.6.1.4.1.89.88.31.1.7.{index-of-acl}.{index-of-rule} i {value of field 3 in
ACL}

```

**Example of adding a deny any any offset-list PADO rule to MAC ACL 7-mac (since the rule is assumed to be the first one, then index-of-rule=20)**

**CLI command:**

```

mac access-list extended 7-mac
deny any any offset-list PADO ace-priority 20
exit

```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.5.1.2.1 i 15 \
1.3.6.1.4.1.89.88.5.1.2.2 i 15 \
1.3.6.1.4.1.89.88.5.1.2.3 i 15 \
1.3.6.1.4.1.89.88.5.1.3.1 i 1 \
1.3.6.1.4.1.89.88.5.1.3.2 i 2 \
1.3.6.1.4.1.89.88.5.1.3.3 i 3 \
1.3.6.1.4.1.89.88.5.1.5.1 i 4 \
1.3.6.1.4.1.89.88.5.1.5.2 i 4 \
1.3.6.1.4.1.89.88.5.1.5.3 i 4

snmpset -v2c -c private 192.168.1.30 \
1.3.6.1.4.1.89.88.31.1.3.207.20 i 2 \
1.3.6.1.4.1.89.88.31.1.4.207.20 i 5 \
1.3.6.1.4.1.89.88.31.1.5.207.20 i 1 \
1.3.6.1.4.1.89.88.31.1.6.207.20 i 2 \
1.3.6.1.4.1.89.88.31.1.7.207.20 i 3
```

### Creating an EtherType-based rule in MAC ACL

**MIB:** qosclimib.mib

**Tables used:**

rIQoSTupleTable — 1.3.6.1.4.1.89.88.5  
rIQoSAceTidxTable — 1.3.6.1.4.1.89.88.31

**Scheme:** a rule is created in two requests.

1. Setting the rule parameters.

```
snmpset -v2c -c <community> <IP address> \
1.3.6.1.4.1.89.88.5.1.2.{value of field 1} i {mac-src(10), mac-dest(11),
vlan(12)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 1} x {protocol index (HEX)} \
1.3.6.1.4.1.89.88.5.1.3.{value of field 1} i {Value in port table for protocol
= 0. Constant for this rule} \
1.3.6.1.4.1.89.88.5.1.2.{value of field 2} i {ether-type(17)} \
1.3.6.1.4.1.89.88.5.1.3.{value of field 2} i {ether-type (DEC)} \
1.3.6.1.4.1.89.88.5.1.4.{value of field 2} x {Zero field is a constant}
```

2. Binding a rule by index-of-rule to an ACL by index-of-acl as permit.

```
snmpset -v2c -c <community> <IP address> \
.1.3.6.1.4.1.89.88.31.1.3.{index-of-acl}.{index-of-rule} i {permit(1)}
.1.3.6.1.4.1.89.88.31.1.4.{index-of-acl}.{index-of-rule} i {mac(5)} \
.1.3.6.1.4.1.89.88.31.1.5.{index-of-acl}.{index-of-rule} i {value of field 1} \
.1.3.6.1.4.1.89.88.31.1.9.{index-of-acl}.{index-of-rule} i {value of field 2}
```

**Example of adding a permit 00:1f:c6:8b:c6:8a 00:00:00:00:00:00 any 806 0000 rule to MAC ACL 7-mac (since the rule is assumed to be the first one, then index-of-rule=20)**

**CLI command:**

```
mac access-list extended 7-mac
 permit 00:1f:c6:8b:c6:8a 00:00:00:00:00:00 any 806 0000 ace-priority 20
exit
```

**SNMP command:**

```
snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.88.5.1.2.1 i 10 \  
1.3.6.1.4.1.89.88.5.1.4.1 x "0x001fc68bc68a00000000000000" \  
1.3.6.1.4.1.89.88.5.1.3.1 i 0 1.3.6.1.4.1.89.88.5.1.2.2 i 17 \  
1.3.6.1.4.1.89.88.5.1.3.2 i 2054 \  
1.3.6.1.4.1.89.88.5.1.4.2 x "0x00 00"

snmpset -v2c -c private 192.168.1.30 \  
1.3.6.1.4.1.89.88.31.1.3.207.20 i 1 \  
1.3.6.1.4.1.89.88.31.1.4.207.20 i 5 \  
1.3.6.1.4.1.89.88.31.1.5.207.20 i 1 \  
1.3.6.1.4.1.89.88.31.1.9.207.20 i 2
```

## TECHNICAL SUPPORT

For technical assistance in issues related to operation of ELTEX Enterprise Ltd. equipment, please contact our Service Center:

Feedback form on the website: <https://eltex-co.ru/support/>

Servicedesk: <https://servicedesk.eltex-co.ru/>

Visit ELTEX official website to get the relevant technical documentation and software, benefit from our knowledge base, send us online request or consult a Service Center Specialist:

Official website: <https://eltex-co.ru/>

Knowledge base: <https://docs.eltex-co.ru/display/EKB/Eltex+Knowledge+Base>

Download center: <https://eltex-co.ru/support/downloads>