SNMP Agent

Teldat Dm712-I

Legal Notice

Warranty

This publication is subject to change.

Teldat offers no warranty whatsoever for information contained in this manual.

Teldat is not liable for any direct, indirect, collateral, consequential or any other damage connected to the delivery, supply or use of this manual.

Table of Contents

Chapter 1 Introduction to the SNMP Protocol

1.1 Introduction

SNMP is an OSI layer 7 (i.e., application layer) protocol used to configure and monitor different router characteristics.

SNMP enables network hosts to read and modify some of the router's operating parameters. Through it, software that runs on a remote host can contact the router over a network and, upon request, get updated information from said router. With this data, it is possible to centrally manage the routers in the network.

SNMP's basic functions include:

- Collecting information and modifying a router's operating parameters for remote SNMP users.
- Sending and receiving SNMP packets via IP.

Protocol Layers of the SNMP Environment

The software used to process SNMP requests is run on the router and is known as the SNMP agent. The user program that builds SNMP requests runs on the user devices elsewhere in the network and not on the router. This is known as the SNMP manager. The SNMP agent, in the router, and the manager, in the work station, use UDP/IP to exchange packets.

For further information on SNMP, please see RFC 1157, *A Simple Network Management Protocol*. Recommendations in RFC 3410 to 3418, RFC 3584 and RFC 3826 all provide information on SNMPv3 (the latest SNMP version).

1.2 SNMP Protocol Versions

There are three versions of SNMP. The first, SNMPv1, defines basic operations, as well as an elemental authentication mechanism based on communities.

SNMPv2 introduced new operations and data types to improve SNMP efficiency. In this second version, efforts were made to improve the protocol's security. However, the complexity of the proposed solution meant, in practical terms, that the SNMPv1 authentication mechanism continued to be used. The SNMPv2 that uses the foregoing authentication mechanism is known as SNMPv2c (where "c" stands for community).

The third version, SNMPv3, set a complete security framework for SNMP. At this point, SNMP was also divided into separate modules to simplify future developments.

1.3 SNMP Packet Types

SNMP supports three basic operations:

- Request data from the agent.
- Configure data in the agent.
- Notifications: information sent by the agent to the manager on any events taking place in the agent.

These three operations are performed in SNMPv1 using the following types of packets:

- GET-REQUEST: the manager uses this command to ask the agent for the state of one or several variables. The manager waits for a response from the agent.
- GET-NEXT: the manager uses this command to ask the agent for the state of the variable that follows the one selected on the management tree. This command is useful when running through tables or to get a subset from the management tree. The manager waits for a response from the agent.
- SET-REQUEST: the manager uses this command to configure the value of one or several variables in the agent. The manager waits for a response from the agent.
- GET-RESPONSE (known simply as RESPONSE from version 2 onwards): the agent uses this command to respond to requests from the manager. In the case of the **get-request** and **get-next** commands, the agent responds with the value of the requested variable. Through the **set-request** command, the agent indicates whether the operation was successful.
- TRAP: the agent uses this command to report a significant event to the manager, but does not expect a response.

New commands were introduced in versions 2 and 3:

- GET-BULK: the manager uses this command to ask the agent for the state of several variables. The definition of

this command makes it especially useful to obtain the state of all the variables in a table.

- INFORM: a manager uses this command to send information about a significant event to another manager. The difference between this and the TRAP command is that the manager is expected to respond with a **response** command.
- NOTIFICATION: this command replaces the TRAP command in version 1.
- REPORT: the agent uses this command to report (to the manager) an unusual error in the treatment of a request made by the manager. This command is used in SNMP version 3 as part of the discovery process for data that must be obtained from the agent.

1.4 Security

1.4.1 SNMPv1 and SNMPv2c

In SNMPv1 and SNMPv2c, the security mechanism is very simple: there is no data encryption and the authentication used is very basic.

Every SNMP packet includes a field indicating the community to which it belongs. Said community is simply a set of characters. In the agent, you can specify the following for each community:

- The access level, regardless of whether it's read or write.
- The view: a set of accessible variables for that community.
- The IP addresses that belong to said community. That is, the IP addresses of the managers that can access the agent using this community.

Each SNMP packet that reaches the router is validated or dropped depending on whether or not it complies with the restrictions imposed by the authentication schema. Specifically, the accessed variable, its type of access and the source IP address of the SNMP packet, must be included in what is associated with the SNMP packet community name.

Given that the community information (in an SNMP packet) is routed in clear, it's very easy to discover the community name and gain access to the agent. This lack of security in SNMPv1 (and, logically, in SNMPv2c) means it is mainly used to monitor the router's status and not, for example, to configure write permissions.

1.4.2 SNMPv3

As mentioned, SNMPv3 has tried to solve the security issues of SNMPv1 and SNMPv2c by defining robust mechanisms for authentication and encryption. Thanks to the modularity in this version, new encryption or authentication algorithms may be added to the protocol. There are currently two authentication mechanisms (HMAC-MD5 and HMAC-SHA) and two encryption mechanisms (DES and AES-128).

In SNMPv3, a security model has been identified. This model has been applied to the SNMP packets and a security level within it.

The currently existing security models are:

- SNMPv1.
- SNMPv2c.
- SNMPv3 USM: User Security Model.

The security levels are as follows:

- noAuthNoPriv: the packets don't use authentication or encryption. It's the only level of security applied to the SNMPv1 and SNMPv2 security model packets.
- AuthNoPriv: the packets use authentication, but are not encrypted.
- AuthPriv: the packets use authentication and encryption.

Note

The use of unauthenticated encryption is not contemplated here.

1.4.3 View-based Access Control

The access control feature determines whether or not an SNMP manager has access to a given variable.

SNMPv1 and SNMPv2 use the community (and perhaps the manager's IP address) to determine whether access to

a specific variable is allowed.

In SNMPv3, the access control mechanism is generalized by introducing more variables in the decision making process. The variables used are as follows:

- Group: this is a set of **<securityName, securityModel>** tuples. The securityName is the SNMP manager identifier (the community in SNMPv1 and SNMPv2, and the user in SNMPv3). All group members have the same access to the agent's variables. A <securityName, securityModel> combination can only belong to one group.
- Context: this is a set of manageable variables accessed by an SNMP entity. A manageable variable can appear in more than one context. For further information on this, please see RFC 3411.
- Security level: access levels can be different for an encrypted and a non-encrypted message. For instance, you can ask for the message to be encrypted before granting write accesses to a variable.

The securityName and the securityModel included in the SNMP packet are used to identify the group. A certain view is displayed depending on the group, context, security model, security level and type of access required (read, write or notification).

A view is merely a set of accessible variables. If the selected view includes the variable you want, access is permitted. If not, access is denied.

Chapter 2 Configuring the SNMP Agent

2.1 Displaying the SNMP Configuration Prompt

Enter **protocol snmp** (general configuration menu) to access the SNMP configuration environment.

```
Config>protocol snmp
-- SNMP user configuration --
SNMP Config>
```

2.2 SNMP Configuration Commands

This section summarizes the SNMP configuration commands.

Command	Function
? (HELP)	Shows the available commands and lists the options associated with specific commands.
ACCESS	Configures the views associated with a specific group and security model.
COMMUNITY	Creates a community or modifies the parameters for an existing community.
CONTEXT	Creates a context or modifies an existing one.
DEFAULT-CONFIG	Enables the default configuration.
DISABLE	Disables SNMP.
ENABLE	Enables SNMP.
ENGINEID	Configures the EngineID used by SNMPv3.
GROUP	Assigns a user and a security model to a group.
HOST	Configures a host to receive SNMP notifications.
LIST	Displays the SNMP configuration.
MIB	Configures MIB options.
NO	Deletes the configuration or sets parameters to their default values.
SUBTREE	Specifies the MIB portions included or excluded from a specific view.
TRAP	Configures notification-sending parameters.
USER	Creates a user or modifies the parameters for an existing user.
EXIT	Exits the SNMP configuration menu.

2.2.1 ? (HELP)

Displays the available commands and their options.

Syntax:

```
SNMP config>?
```

Example:

```
SNMP config>?
  access            Specify access views associated with a group and security
                    model
  community         Create a community or modify parameters of an existing one
  context           Create a context or modify an existing one
  default-config    Enable the SNMP default configuration
  disable           Disable SNMP
  enable            Enable SNMP
  engineid          Specify an SNMPv3 EngineID
  group             Assign a <user, securityModel> tuple to a group
  host              Specify a host to receive SNMP notification messages
  list              Display SNMP configuration
  mib               Configure options for a MIB
  no                Negate a command or set its defaults
```

```
  subtree           Create or modify a MIB view
  trap              Set trap parameters
  user              Create a SNMPv3 user or modify parameters of an existing one
  exit              Exit SNMP configuration menu
SNMP config>
```

2.2.2 ACCESS

Configures the views associated with a specified group and security model. Assigns a read, write and notification view to a specific group and security model. You can also configure different views for different security models (within the same group), and different views for the same group using different contexts.

Note

To obtain information from a secondary VRF through a context, you can only submit the following requests: over the MIB IP-FORWARD-MIB and, more specifically, over the ipCidrRouteNumber object (OID: 1.3.6.1.2.1.4.24.3) and the ipCidrRouteTable table (OID: 1.3.6.1.2.1.4.24.4), and over the MIB2 ipAddrTable (OID: 1.3.6.1.2.1.4.20).

To obtain information from the main VRF, submit the request as normal (even if this is done from a manager that is in the secondary VRF).

Please see *View-based Access Control* on page 2 for further information on access control based on views.

Syntax:

```
SNMP config>group <GroupName> [context <ContextName>] <securityModel> <securityLevel>
{read-view <view> | notify-view <view> | write-view <view>}
```

securityModel: This can be:

- *any*: any security model.
- *v1*: SNMPv1.
- *v2c*: SNMPv2c.
- *v3-usm*: SNMPv3 User Security Model (currently this is the only security model defined for SNMPv3).

securityLevel: (only if you select v3-usm as the security model). This can be:

- *auth*: authentication, no encryption.
- *noauth*: no authentication, no encryption.
- *authpriv*: authentication and encryption.

Note

If no view has been specified, the view associated with all OIDs is default. This view is always created and known internally as "_all_". If you do not want to access all OIDs, specify the "_none_" view.

Example:

Associating a *teldatreadview* read view and a *teldatnotifyview* notification view with the *teldatgroup* for SNMPv3 with encryption and authentication.

```
SNMP config>access teldatgroup v3-usm priv read-view teldatreadview notify-view
 teldatnotifyview
SNMP config>
```

Example:

Associating a *teldatwriteview* write view with the *teldatgroup2* group for SNMPv1.

```
SNMP config>access teldatgroup2 v1 write-view teldatwriteview
SNMP config>
```

2.2.3 COMMUNITY

Creates a community or alters the parameters of a community that already exists.

Syntax:

```
SNMP config>community <Community Name>
  default    Create a SNMP community with default values
  access     Set access permissions for this community
  context    Set a context for this community
  subnet     Specify subnet with access using this community string
  view       Set a view for this community
SNMP config>
```

Community Name	Specifies the name of the community (32 characters maximum). Special characters such as spaces, tabs, and so on, are not accepted.

2.2.3.1 COMMUNITY community_name DEFAULT

Creates a community with the default parameters or resets said parameters for a community that already exists. The default parameters are as follows:

- Read access and generation of traps.
- View associated with all MIBs.
- Access allowed from all IP addresses.

Example:

```
SNMP config>community public default
SNMP config>
```

2.2.3.2 COMMUNITY community_name ACCESS

Establishes the access level associated with a community. The possible access levels are as follows:

read-trap: Read and trap generation.

trap-only: Trap generation.

write-read-trap: Read-write and trap generation.

Syntax:

```
SNMP config>community public access ?
  read-trap          Read SNMP variables and generate traps
  trap-only          Generate traps only
  write-read-trap    Read and write SNMP variables and generate traps
SNMP config>
```

The read access and generation of traps are allowed by default.

Example:

```
SNMP config>community public access write-read-trap
SNMP config>
```

2.2.3.3 COMMUNITY community_name CONTEXT

Specifies a context to be assigned to a specific community. This context lets you create an association between a community and a secondary VRF.

Note

To obtain information from a secondary VRF through a context, you can only execute the following requests: over the MIB IP-FORWARD-MIB and, more specifically, over the ipCidrRouteNumber object (OID: 1.3.6.1.2.1.4.24.3) and the ipCidrRouteTable table (OID: 1.3.6.1.2.1.4.24.4), and over the MIB2 ipAddrTable (OID: 1.3.6.1.2.1.4.20).

To gather information from the main VRF, submit the request as normal (even if this is done from a manager that is in the secondary VRF).

Syntax:

```
SNMP config>community public context ?
```

```
  <1..32 chars>    Context name
SNMP config>
```

Example:

```
SNMP config>community public context cntxt
SNMP config>
```

2.2.3.4 COMMUNITY community_name SUBNET

Specifies a subnet where access is allowed using a specified community

Note

SNMP requests may arrive for any of the router's addresses.

You can specify more than one subnet for a community. To do this, repeat the operation for each additional subnet you want to add.

SNMP requests are accepted for a community (in some pre-configured addresses) if the outcome of the AND function between the IP address that originates the request and the community network mask matches the outcome of the AND function between the community IP address and its mask (i.e., requests are accepted from any device in the subnets defined by the masks). If no address is specified for the community, requests are accepted from any host.

Syntax:

```
SNMP config>community public subnet ?
  <a.b.c.d>    IP Address
SNMP config>community public subnet 192.6.2.168 ?
  <a.b.c.d>    Mask
SNMP config>community public subnet 192.6.2.168 255.255.255.0 ?
  <cr>
SNMP config>
```

Access is granted from any IP address by default.

Example 1:

```
SNMP config>community public subnet 192.6.2.168 255.255.255.0
SNMP config>
```

Or:

The aim of this operation is to ensure that *public* community requests are accepted when they come from any host on the 192.6.2.0 network.

Example 2:

```
SNMP config>community public subnet 192.6.2.168 255.255.255.255
SNMP config>
```

The aim of this operation is to ensure that *public* community requests are accepted only when they come from the 192.6.2.168 host.

2.2.3.5 COMMUNITY community_name VIEW

Assigns an MIB view to a community. This view must be pre-created using the **subtree** command. If the view associated with the community doesn't exist, access is restricted to all MIBs. If the view associated with a community is **all,** the community can access all MIBs.

Example:

```
SNMP config>community private view teldat
SNMP config>
```

Default is access allowed to all MIBs.

2.2.4 CONTEXT

Creates a context or modifies an existing one.

Syntax:

```
SNMP config>context <Context-Name>
```

Example:

```
SNMP config>context teldatContext
SNMP config>
```

2.2.5 DEFAULT-CONFIG

Enables default configuration. **default-config** enables SNMP and creates a community called “teldat”, with the following characteristics: it has all permissions (read, write and trap generation), accepts requests from any address, and has a complete MIB view.

The default configuration is enabled by default.

Syntax:

```
SNMP config>default-config
```

2.2.6 DISABLE

Disables SNMP.

Syntax:

```
SNMP config>disable
```


Note

If the default configuration is enabled by default, SNMP is always enabled. This means SNMP cannot be disabled until the default configuration is disabled.

2.2.7 ENABLE

Enables SNMP.

Syntax:

```
SNMP Config>enable
```

Default is SNMP enabled.

2.2.8 ENGINED LOCAL

Configures the engineID the router uses in SNMPv3. The engineID uniquely identifies an SNMP entity when it uses SNMPv3.

Syntax:

```
SNMP config>engineid local ?
  <8..64 hex chars>    EngineID hexadecimal octet string
SNMP config>
```

Default is no engineID configured (a random engineID is generated on device startup).

2.2.9 GROUP

Assigns a user and a security model to a group. The **<user, security model>** tuple can only belong to one group. The groups use this to define access control based on views. For further information, please see *View-based Access Control* on page 2.

Syntax:

```
SNMP config>group <secName> <secModel> <groupName>
```

secName: user going to be added to a group.

securityModel: security model. This can be:

- *any*: any security model.
- *v1*: SNMPv1.
- *v2c*: SNMPv2c.
- *v3-usm*: SNMPv3 User Security Model (currently this is the only security model defined for SNMPv3).

groupName: group name

Example:

Creating the *teldatgroup* group, with *teldatuser* users for SNMPv3 and *teldatuser1* for SNMPv1.

```
SNMP config>group teldatuser v3-usm teldatgroup
SNMP config>group teldatuser2 v1 teldatgroup
SNMP config>
```

2.2.10 HOST

Configures a host to receive SNMP notifications. This includes the sending characteristics for these notifications.

Syntax:

```
SNMP config>host {<IPv4 address> | <hostname>} {trap | inform} version <secModel> <secLevel> <secName>
[udp-port <port>] [vrf <vrf-name>] [{traps}]
SNMP config>
```

IPv4 address: host IP address SNMP notifications are sent to.

hostname: name used to resolve the IP address where SNMP notifications are sent.

trap: notifications are sent as TRAPS.

inform: notifications are sent as INFORMs.

secModel: specifies the SNMPv3 version used to send the notifications. Possible versions are:

- v1: SNMPv1 (not available for *inform*).
- v2c: SNMPv2c.
- v3: SNMPv3.

secLevel: security level used when sending notifications. This is only configurable for SNMPv3. Possible values are:

- *auth*: authentication, no encryption.
- *noauth*: no authentication, no encryption.
- *authpriv*: authentication and encryption.

secName: user or community name used when sending notifications.

udp-port <port>: UDP port notifications are sent to. This parameter is optional. If it isn't specified, notifications are sent to port 162.

vrf <vrf-name>: secondary VRF over which notifications are sent. This parameter is optional. If it isn't specified, notifications are sent over the main VRF.

traps: specifies the type of notifications sent to the host. You can specify various types of notifications. These are as follows:

Type	Description
all	All notifications.
authentication-failure	Notifications for authentication failures.
cold-start	Notifications when the router has executed a cold-start.
enterprise-specific	Enterprise-specific notifications. Enterprise-specific notifications focus on an event defined as unusual and should be reported. The *specific-trap* field identifies the specific event that has occurred. In our router, enterprise-specific notifications are those configured as such in the Events Logging System. On enabling specific notifications, the rest of the groups automatically enable. Consequently, events enabled for snmp-trap and for all the remaining groups reach the host.
enterprise-specific-group1	Enterprise-specific notifications group 1. Enterprise-specific notifications focus on

	an event defined as unusual and should be reported. The notification *specific-trap-group1* field identifies the specific event. In our router, enterprise-specific notifications are those configured as such in the Events Logging System (ELS).
enterprise-specific-group2	Enterprise-specific notifications group 2. Enterprise-specific notifications focus on an event defined as unusual and should be reported. The notification *specific-trap-group2* field identifies the specific event. In our router, enterprise-specific notifications are those configured as such in the Events Logging System (ELS).
enterprise-specific-group3	Enterprise-specific notifications group 3. Enterprise-specific notifications focus on an event defined as unusual and should be reported. The notification *specific-trap-group3* field identifies the specific event. In our router, the enterprise-specific notifications are those configured as such in the Events Logging System (ELS).
enterprise-specific-group4	Enterprise-specific notifications group 4. Enterprise-specific notifications focus on an event defined as unusual and should be reported. The notification *specific-trap-group4* field identifies the specific event. In our router, the enterprise-specific notifications are those configured as such in the Events Logging System (ELS).
link-down	Link-down notification. Reports an error in one of the router interfaces. Said notification contains the *ifIndex* value for the interface affected as the first element in its variables list.
link-up	Link-up notification. Notifies that the down interface is now up and operating again. Said notification contains the *ifIndex* value for the interface affected as the first element in its variables list.
warm-start	Notifications when the router has executed a warm-start.

Example:

The host is configured in 172.24.51.12 address to send encrypted SNMPv3 traps using the *teldatuser*. Said notifications are sent to port 170. Notifications are sent to report changes in the state of the interface.

```
SNMP config>host 172.24.51.12 trap version v3 auth teldatuser udp-port 170 link-
up link-down
SNMP config>
```

Example:

The host is configured with the "host0.sample.es" hostname to send SNMPv2 reports using *teldatcomm1.* All notifications are sent.

```
SNMP config>host host0.sample.es inform version v2 teldatcomm1 all
SNMP config>
```

Example:

The host is configured in address 172.24.51.12 to send SNMPv1 traps using *teldatcomm2.* Enterprise specific notifications are sent.

```
SNMP config>host 172.24.51.12 trap version v1 teldatcomm2 enterprise-specific
SNMP config>
```

Command history:

Release	Modification
11.01.04	The *hostname* option has been added.

2.2.11 LIST

Lists the SNMP configuration.

Syntax:

```
SNMP config>list ?
  access                      Display current access views configuration
  all                         Display all the SNMP configuration information
  community                   Display current communities configuration
  group                       Display current groups configuration
  host                        Display current hosts configuration
  trap-sending-parameters     Display trap sending configuration
  user                        Display current users configuration
  view                        Display current views configuration
SNMP config>
```

2.2.11.1 LIST ACCESS

Lists the views configured for each group and security model.

Example:

```
SNMP config>list access
--------------
Access Entries
--------------
Group Name    Context    secModel   secLevel           Views
------------  ----------  -------- ------------  ------------------------------
teldatgroup2              v1       noAuthNoPriv  readView:
                                                 writeView: teldatwriteview
                                                 notifyView:
teldatgroup               v3-usm   Priv          readView: _all_
                                                 writeView: _all_
                                                 notifyView: _all_
pap           sd          any      AuthNoPriv    readView: fd
                                                 writeView: ff
                                                 notifyView: df
teldatgroup               v3-usm   Priv          readView: teldatreadview
                                                 writeView:
                                                 notifyView: teldatnotifyviewe
SNMP config>
```

2.2.11.2 LIST ALL

Displays all the SNMP configuration information.

Example:

```
SNMP config>list all
Default configuration is disabled
SNMP is enabled
No trap sending parameters, showing default values

Max time storing traps (seg):                   50
Max number of traps to store:                   32
Manager reachability check before sending traps: UDP echo

        Community Name              IP Address       IP Mask
--------------------------------   --------------------------
public                             ALL
private                            192.6.2.168      255.255.255.255

        Community Name              Access
--------------------------------   --------------------------
public                             Read, Trap
private                            Read, Write, Trap

        Community name              Views
--------------------------------   --------------------------
public                             mib2
private                            teldat

        View name                   Subtree
--------------------------------   --------------------------
mib2                               1.3.6.1.2.1 (included)
teldat                             1.3.6.1.4.1.2007 (included)

--------------
Access Entries
--------------
Group Name    Context    secModel   secLevel           Views
------------  ----------  -------- ------------  ------------------------------
teldatgroup2              v1       noAuthNoPriv  readView:
                                                 writeView: teldatwriteview
```

```
                                          notifyView:

teldatgroup              v3-usm   Priv          readView: _all_
                                               writeView: _all_
                                               notifyView: _all_
pap           sd         any      AuthNoPriv    readView: fd
                                               writeView: ff
                                               notifyView: df
teldatgroup              v3-usm   Priv          readView: teldatreadview
                                               writeView:
                                               notifyView: teldatnotifyviewe

------
Groups
------
SecName               secModel  Group name
--------------------  --------  -------------------
teldat                v3-usm    teldatgroup2
teldatmd5aes          v3-usm    teldatgroup2
teldatmd5des          v3-usm    teldatgroup2
teldatshades          v3-usm    teldatgroup2
teldatuser            v3-usm    teldatgroup2
teldatuser2           v1        teldatgroup

-----
Hosts
-----
SecName       Hostname/IP Address      Type  Port    VRF Name       Security             Traps
------------ ------------------------- ------ ----- ------------ -------------------- --------------------
teldat      172.24.51.12              trap   162   <main>       v3-usm Priv          Cold Start
                                                                                     Warm Start
                                                                                     Link Down
                                                                                     Link Up
                                                                                     Auth. Failure
                                                                                     Enterprise Specific
teldat2     172.24.51.12              trap   162   <main>       v3-usm AuthNoPriv    Cold Start
                                                                                     Warm Start
                                                                                     Link Down
                                                                                     Link Up
                                                                                     Auth. Failure
                                                                                     Enterprise Specific
teldatcomm2 172.24.51.12              trap   162   <main>       v1     noAuthNoPriv  Enterprise Specific
teldatuser  host0.sample.es           trap   170   <main>       v3-usm AuthNoPriv    Link Down
                                                                                     Link Up
                                                                                     Ent. Sp. Group 2
                                                                                     Ent. Sp. Group 3
teldatcomm1 172.24.51.12              inform 162   <main>       v2c    noAuthNoPriv  None

-----
Users
-----
              SecName            auth type  priv type
-------------------------------  ---------  ---------
teldat                           sha        aes-128
teldatmd5aes                     md5        aes-128
teldatmd5des                     md5        des
teldatshades                     sha        des
teldatuser                       sha        des
teldatuser2                      sha        aes-128

SNMP config>
```

Note

If the default configuration is enabled, SNMP is always enabled.

Command history:

Release	Modification
11.01.04	The *hostname* option now appears in the list of SNMP hosts.

2.2.11.3 LIST COMMUNITY

Lists information on the configured communities.

Syntax:

```
SNMP config>list community ?
  access     Display the access mode information for all communities
  address    Display the associated addresses information for all communities
  view       Display the view information associated to each community
SNMP config>
```

LIST COMMUNITY ACCESS

Lists information on the access level associated with the different configured communities.

Example:

```
SNMP config>list community access
        Community Name              Access
--------------------------------   -------------------------
public                              Read, Trap
private                             Read, Write, Trap
SNMP config>
```

LIST COMMUNITY ADDRESS

Lists information on subnets that are authorized to use the different communities.

Example:

```
SNMP config>list community address
        Community Name              IP Address       IP Mask
--------------------------------   -------------------------
public                              ALL
private                             192.6.2.168      255.255.255.255
SNMP config>
```

LIST COMMUNITY VIEW

Lists view information associated with each community.

Example:

```
SNMP config>list community view
        Community name              Views
--------------------------------   -------------------------
public                              mib2
private                             teldat
SNMP config>
```

2.2.11.4 LIST GROUP

Lists information on the configured groups.

Example:

```
SNMP config>list group
------
Groups
------
SecName               secModel  Group name
--------------------  --------  -------------------
teldat                v3-usm    teldatgroup2
teldatmd5aes          v3-usm    teldatgroup2
```

```
teldatmd5des          v3-usm    teldatgroup2
teldatshades          v3-usm    teldatgroup2
teldatuser            v3-usm    teldatgroup2
teldatuser2           v1        teldatgroup
SNMP config>
```

2.2.11.5 LIST HOST

Lists information on the configured notifications.

Example:

```
SNMP config>list host

-----
Hosts
-----
SecName       Hostname/IP Address      Type  Port    VRF Name       Security             Traps
----------- ------------------------- ------ ----- ------------ ------------------- -------------------
teldat      172.24.51.12              trap   162   <main>         v3-usm Priv       Cold Start
                                                                                    Warm Start
                                                                                    Link Down
                                                                                    Link Up
                                                                                    Auth. Failure
                                                                                    Enterprise Specific
teldat2     172.24.51.12              trap   162   <main>         v3-usm AuthNoPriv Cold Start
                                                                                    Warm Start
                                                                                    Link Down
                                                                                    Link Up
                                                                                    Auth. Failure
                                                                                    Enterprise Specific
teldatcomm2 172.24.51.12              trap   162   <main>         v1  noAuthNoPriv  Enterprise Specific
teldatuser  host0.sample.es           trap   170   <main>         v3-usm AuthNoPriv Link Down
                                                                                    Link Up
                                                                                    Ent. Sp. Group 2
                                                                                    Ent. Sp. Group 3
teldatcomm1 172.24.51.12              inform 162   <main>         v2c  noAuthNoPriv None
SNMP config>
```

Command history:

Release	Modification
11.01.04	The *hostname* option now appears in the list of SNMP hosts.

2.2.11.6 LIST TRAP-SENDING-PARAMETERS

Lists information concerning notification sending.

Example:

```
SNMP config>list trap sending-parameters
No trap sending parameters, showing default values

Max time storing traps (seg):                    50
Max number of traps to store:                    32
Manager reachability check before sending traps: UDP echo

SNMP config>
```

2.2.11.7 LIST USER

Lists information on the configured users.

Example:

```
SNMP config>list user
-----
Users
```

```
-----
              SecName            auth type  priv type
------------------------------  ---------  ---------
teldat                          sha        aes-128
teldatmd5aes                    md5        aes-128
teldatmd5des                    md5        des
teldatshades                    sha        des
teldatuser                      sha        des
teldatuser2                     sha        aes-128
SNMP config>
```

2.2.11.8 LIST VIEW

Lists information on the views defined in the system, together with the MIB portions or *subtrees* associated with each.

Example:

```
SNMP config>list view
        View name                   Subtree
---------------------------------   -------------------------
teldatview                          1.3.6.1.4.1.2007 (included)
mib2                                1.3.5.1.2.1 (included)
SNMP config>
```

2.2.12 MIB

Configures MIB options.

Syntax:

```
SNMP config>mib <mibName> <variableName> <option>
```

2.2.12.1 EVENTS

Configures the options under *EVENTS* for Teldat's MIB.

Syntax:

```
SNMP config>mib ?
  ifmib    Options for IF-MIB
  events    Events in proprietary MIB
```

Command history:

Release	Modification
11.01.06	The *events* option has been added.

2.2.12.1.1 MIB EVENTS DISABLE

Allows events to be disabled from Teldat's MIB.This is useful for snmp operations requested through a graphical mib-browser.

```
SNMP config>mib events ?
  disable    Disable MIB events in SNMP operations
SNMP config>mib events disable
```

Example:

```
SNMP config>mib events disable
```

Command history:

Release	Modification
11.01.06	The *disable* suboption was added under *events*.

2.2.12.2 MIB IFMIB

Configures options for the *IF-MIB* MIB.

Syntax:

```
SNMP config>mib ?
  ifmib    Options for IF-MIB
```

2.2.12.2.1 MIB IFMIB IFALIAS

Configures options for the *ifAlias* variable that belongs to the *IF-MIB* MIB.

```
SNMP config>mib ifmib ?
  ifalias    Options for ifAlias variable
SNMP config>mib ifmib ifalias ?
  256    Change ifAlias value larger than 64 characters
```

- **256:** Increases the maximum length allowed for the the character string constituting the value of the *ifAlias* variable.

Example:

```
SNMP config>mib ifmib ifalias 256
```

2.2.13 NO

Configures the parameters with their default values or deletes the configuration.

Syntax:

```
SNMP config>no ?
  access             Specify access views associated to a group and security
                     model
  community          Create a community or modify parameters of an existing one
  context            Create a context or modify an existing one
  default-config     Enable the SNMP default configuration
  engineid           Specify an SNMPv3 EngineID
  group              Assign a <user, securityModel> tuple to a group
  host               Specify a host to receive SNMP notification messages
  mib                Configure options for a MIB
  subtree            Create or modify a MIB view
  trap               Set trap parameters
  user               Create a SNMPv3 user or modify parameters of an existing one
SNMP config>
```

Example:

Deleting the *private* community.

```
SNMP config>no community private
SNMP config>
```

Example:

Deleting all notification configuration for host 172.24.51.12.

```
SNMP config>no host 172.24.51.12
SNMP config>
```

Example:

Default configuration for the maximum time a notification is kept before being sent.

```
SNMP config>no trap sending-parameters time
SNMP config>
```

2.2.14 SUBTREE

Creates a new view or adds a portion of the MIB to a view that already exists. Part of an MIB can be included or excluded from the existing view.

To assign a view to one or more communities, run the **community community_name view** command and use the **access** command to assign a view to a group.

Syntax:

```
SNMP config>subtree <viewName> <OID> <included | excluded>
SNMP config>
```

View name	Specifies the name of the view (32 characters maximum). Special characters such as spaces, tabs, and so on, are not accepted.
OID	Specifies the MIB Object ID for the *subtree.* All objects that hang off this OID are included in, or excluded from, the view.
included	The specified OID is included in the view (it can be accessed).
excluded	The specified OID is excluded from the view (it can't be accessed).

Example:

```
SNMP config>subtree mib2 1.3.6.1.2.1 included
SNMP config>
```

Note

A view does not show any *excluded* portions of a MIB. In fact, access is restricted, by default, to all variables that are not explicitly specified as *included.*

2.2.15 TRAP

Configures the parameters used to set trap-sending conditions.

Syntax:

```
SNMP config>trap ?
  sending-parameters    Set trap sending parameters
SNMP config>
```

2.2.15.1 TRAP SENDING-PARAMETERS

Configures the trap-sending parameters. SNMP trap-sending triggers X.25 or ISDN calls if the destination for said trap is on the other side of an interface of this type. Traps should be grouped in a buffer and sent together to reduce the number of calls. It's also a good idea to check that the trap destination is reachable (i.e., a call is established, as in the previous example), which reduces the risk of traps being lost along the route. If, however, you want to receive the traps as soon as possible, minimize the number of traps to save in said buffer before being sent or adjust the maximum wait time for trap-sending. In the latter case, we don't recommend checking the reachability of the manager station as this can cause a certain delay (waiting for a response to the ECHO UDP or ICMP, for example).

Another parameter that can be configured through this command is the type of traps sent when an event is reported.

Trap-sending parameters configured through this option are:

NUMBER	Size of the trap buffer to regroup: number of traps that can be stored before being sent to their destination.
REACHABILITY-CHECKING	Indicates if reachability checks for the manager stations (configured as trap destination) should be carried out before sending traps.
TIME	Maximum time a trap is stored in the buffer before being sent (provided the buffer has not reached its maximum capacity).
FORMAT	Format of the *variable bindings* list, built when a trap is sent.

Syntax:

```
SNMP config>trap sending-parameters ?
  number                    Number of traps to store before sending
  reachability-checking     Reachability checking before sending traps
  time                      Max time to store traps in buffer before sending
  format                    Configure the variable-bindings list format
SNMP config>
```

TRAP SENDING-PARAMETERS NUMBER

Configures the size of the trap buffer to regroup (i.e., the number of traps that can be stored before being sent to their destination). In all cases, traps are sent individually in UDP packets.

Syntax:

```
SNMP config>trap sending-parameters number ?
  <1..63>    Max number of traps to store
SNMP config>
```

Default is 32 traps stored.

Example:

```
SNMP config>trap sending-parameters number 30
SNMP config>
```

TRAP SENDING-PARAMETERS REACHABILITY-CHECKING

This parameter indicates if reachability checking for manager stations (configured as trap destinations) is executed before sending. The values allowed are:

icmp: enables ECHO ICMP sending to destinations to check if these are accessible.

ip-route: traps are only sent when a route to the destination has existed for more than 10 seconds (or for the configured value). Through the **up-delay** option, you can configure the time an IP route (to the destination) must exist to determine if it is accessible.

udp: enables ECHO UDP sending to destinations to check if these are accessible.

Syntax:

```
SNMP config>trap sending-parameters reachability-checking ?
  icmp         ICMP checking
  ip-route     IP route checking
  udp          UDP checking
SNMP config>trap sending-parameters reachability-checking ip-route ?
  up-delay     IP route uptime to consider it accessible
  <cr>
SNMP config>trap sending-parameters reachability-checking ip-route up-delay ?
  <1s..10m>    Route uptime
SNMP config>
```

Default is ECHO UDP sending.

Example:

```
SNMP config>trap sending-parameters reachability-checking ip-route up-delay 15s
SNMP config>
```

TRAP SENDING-PARAMETERS TIME

Maximum time a trap is stored in the buffer before it is sent (assuming the buffer isn't full). I.e., traps are sent when the buffer is full or when the maximum seconds parameter has timed out.

Syntax:

```
SNMP config>trap sending-parameters time ?
  <0s..3550w>    Max time storing traps (Time value)
SNMP config>
```

Default: traps are stored for a maximum of 50 seconds.

Example:

```
SNMP config>trap sending-parameters time 40s
SNMP config>
```

TRAP SENDING-PARAMETERS FORMAT

The type of trap generated when an event is reported depends on the format configured through this command.

With the *legacy* option, each system event reported using SNMP becomes a specific type of trap for this event. This includes a *variable-bindings* list, whose size depends on the number of event arguments. When two traps of this type are received, the same *variable-binding* in each of the traps can contain object instances whose meaning and syntax differ depending on the event that generates them. This mode of sending traps is proprietary and does not follow the standard.

Through the *standard* format option, each notified event is transformed into a specific trap for said event. Arguments

for this event are converted in the *variable-bindings* list defined for this type of trap.

Syntax:

```
SNMP config> trap sending-parameters format { legacy | standard }
```

Default is legacy.

2.2.16 USER

Configures a user for SNMPv3. Specifies the authentication and encryption characteristics associated with said user.

Syntax:

```
SNMP config>user <userName> [auth {md5|sha} {plain|ciphered} <auth-key> [priv {aes128|des}
{plain|ciphered} <priv-key>]]
```

userName:	User name.
auth:	Type of authentication used: *md5* or *sha*.
plain:	Enters the key in clear.
ciphered:	Enters the encrypted key.
auth-key:	Key used for authentication.
priv:	Type of authentication used: *aes128* or *des*.
priv-key:	Key used for encryption.

Example:

Configuration for a teldat user without security parameters (without authentication or encryption).

```
SNMP config>user teldat
```

Example:

Configuration for a *teldatmd5* user with MD5 authentication using the *teldatauth* key and without encryption.

```
SNMP config>user teldatmd5 auth md5 plain teldatauth
```

Example:

Configuration for a *teldatshaaes* user with SHA authentication using the *teldatauth2* key and with AES-128 encryption using the *teldatpriv*key.

```
SNMP config>user teldatshaaes auth sha plain teldatauth2 priv aes128 plain teldatpriv
```

2.2.17 EXIT

Returns to the configuration prompt.

Syntax:

```
SNMP Config>exit
```

Example:

```
SNMP Config>exit
Config>
```

Chapter 3 Monitoring the SNMP Agent

3.1 Accessing the SNMP Monitoring Environment

Enter the SNMP monitoring environment by running the **SNMP protocol** command (general monitoring menu).

```
+protocol snmp
-- SNMP protocol monitor --
SNMP+
```

3.2 SNMP Monitoring Commands

Command	Function
? (HELP)	Displays the commands and their options.
LIST	Displays information on the SNMP protocol.
EXIT	Exits the SNMP monitoring menu.

3.2.1 ? (HELP)

Displays the available commands and their options.

Syntax:

```
SNMP+?
```

Example:

```
SNMP+?
  list    Show protocol information
  exit
SNMP+
```

3.2.2 LIST

Lists the current configuration for SNMP.

Syntax:

```
SNMP+list ?
  access-entry              Access entries information
  all                       All the information
  community                 Communities information
  debug                     Debug information
  group                     Groups information
  host                      Hosts and notifications information
  statistics                SNMP statistics
  trap-sending-parameters   Information related to sending traps
  user                      Users information
  view                      Views defined in the system
```

3.2.2.1 LIST ACCESS-ENTRY

Lists the views associated with each group and security model.

Example:

```
SNMP+list access-entry
          Access Group Name         Access Parameters
--------------------------------   -------------------------
grpcomm1
                                   Context:
                                   Context match: prefix
                                   Security model: any noAuthNoPriv
```

```
                                    Read view: teldatview
                                    Write view: teldatview
                                    Notify view: teldatview
                                    Storage Type: permanent
                                    Row status: active

grpcomm2
                                    Context:
                                    Context match: prefix
                                    Security model: any noAuthNoPriv
                                    Read view: _all_
                                    Write view: <not specified>
                                    Notify view: _all_
                                    Storage Type: permanent
                                    Row status: active

teldatgroup
                                    Context:
                                    Context match: exact
                                    Security model: v3 (USM) authPriv
                                    Read view: _all_
                                    Write view: _all_
                                    Notify view: _all_
                                    Storage Type: permanent
                                    Row status: active

teldatgroup2
                                    Context:
                                    Context match: exact
                                    Security model: v1 noAuthNoPriv
                                    Read view: _all_
                                    Write view: teldatwriteview
                                    Notify view: _all_
                                    Storage Type: permanent
                                    Row status: active
```

Note

The configuration of communities is internally transformed into that of groups. As a result, groups that have not been explicitly configured appear.

3.2.2.2 LIST ALL

Lists all the information for the SNMP configuration that is currently active.

Syntax:

```
SNMP+list all
```

Example:

```
SNMP+list all
SNMP is enabled

Max time storing traps (seg):                   50
Max number of traps to store:                   32
Manager reachability check before sending traps: UDP echo

        Community Name              IP Address       IP Mask
--------------------------------   --------------------------
private                             ALL

        Community Name              Access
--------------------------------   --------------------------
private                             Read, Write, Trap

        Community name              Views
```

```
---------------------------------   -------------------------
private                             teldatview

        Community Name              SecName Parameters
---------------------------------   -------------------------
private                             Context:
                                    SecName: comm1
                                    Network prefix: 0.0.0.0/0

          Access Group Name         Access Parameters
---------------------------------   -------------------------
grpcomm1
                                    Context:
                                    Context match: prefix
                                    Security model: any noAuthNoPriv
                                    Read view: teldatview
                                    Write view: teldatview
                                    Notify view: teldatview
                                    Storage Type: permanent
                                    Row status: active

grpcomm2
                                    Context:
                                    Context match: prefix
                                    Security model: any noAuthNoPriv
                                    Read view: _all_
                                    Write view: <not specified>
                                    Notify view: _all_
                                    Storage Type: permanent
                                    Row status: active

teldatgroup
                                    Context:
                                    Context match: exact
                                    Security model: v3 (USM) authPriv
                                    Read view: _all_
                                    Write view: _all_
                                    Notify view: _all_
                                    Storage Type: permanent
                                    Row status: active

teldatgroup2
                                    Context:
                                    Context match: exact
                                    Security model: v1 noAuthNoPriv
                                    Read view: _all_
                                    Write view: teldatwriteview
                                    Notify view: _all_
                                    Storage Type: permanent
                                    Row status: active

          Group Name                Parameters
---------------------------------   -------------------------
grpcomm1                            SecName: comm1
                                    Security model: v1
                                    Storage Type: permanent
                                    Row status: active

teldatgroup                         SecName: teldatuser2
                                    Security model: v1
                                    Storage Type: permanent
                                    Row status: active

grpcomm1                            SecName: comm1
                                    Security model: v2c
                                    Storage Type: permanent
                                    Row status: active
```

```
teldatgroup                       SecName: teldatuser
                                  Security model: v3 (USM)
                                  Storage Type: permanent
                                  Row status: active


There are no host notifications

          User Name               Parameters
--------------------------------  --------------------------
teldatuser
                                  Engine ID: 1234567890
                                  Group-name: teldatgroup
                                  SecName: teldat
                                  Authentication Protocol: SHA
                                  Privacy Protocol: AES128

_all_
                                  view subtree: .0
                                  view mask:
                                  view type: included
                                  Storage Type: permanent
                                  Row status: active
_all_
                                  view subtree: .1
                                  view mask:
                                  view type: included
                                  Storage Type: permanent
                                  Row status: active
_all_
                                  view subtree: .2
                                  view mask:
                                  view type: included
                                  Storage Type: permanent
                                  Row status: active
_none_
                                  view subtree: .0
                                  view mask:
                                  view type: excluded
                                  Storage Type: permanent
                                  Row status: active
_none_
                                  view subtree: .1
                                  view mask:
                                  view type: excluded
                                  Storage Type: permanent
                                  Row status: active
_none_
                                  view subtree: .2
                                  view mask:
                                  view type: excluded
                                  Storage Type: permanent
                                  Row status: active
SNMP+
```

3.2.2.3 LIST COMMUNITY

Lists information on the configured communities.

Syntax:

```
SNMP+list community ?
  access     Display the access mode information for all communities
  address    Display the associated addresses information for all communities
  view       Display the view information associated to each community
```

LIST COMMUNITY ACCESS

Lists information on the access levels associated with the different configured communities.

Syntax:

```
SNMP+list community access
```

Example:

```
SNMP+list community access
        Community Name              Access
--------------------------------   -------------------------
public                             Read, Trap
private                            Read, Write, Trap
SNMP+
```

LIST COMMUNITY ADDRESS

Lists information on the subnets that can be accessed through the different communities.

Syntax:

```
SNMP+list community address
```

Example:

```
SNMP+list community address
        Community Name              IP Address       IP Mask
--------------------------------   -------------------------
public                             ALL
private                            192.6.2.168      255.255.255.255
SNMP+
```

LIST COMMUNITY VIEW

Lists information on the views associated with each community.

Syntax:

```
SNMP+list community view
```

Example:

```
SNMP+list community view
        Community name              Views
--------------------------------   -------------------------
public                             mib2
private                            teldat
SNMP+
```

3.2.2.4 LIST DEBUG

Lists information on the internal structures used in the SNMP code. This information is only useful for debugging purposes.

3.2.2.5 LIST GROUP

Lists information on the configured groups.

Syntax:

```
SNMP+list group
```

Example:

```
SNMP+list group
          Group Name                Parameters
--------------------------------   -------------------------
grpcomm1                           SecName: comm1
                                   Security model: v1
                                   Storage Type: permanent
```

```
                                  Row status: active

teldatgroup                       SecName: teldatuser2
                                  Security model: v1
                                  Storage Type: permanent
                                  Row status: active

grpcomm1                          SecName: comm1
                                  Security model: v2c
                                  Storage Type: permanent
                                  Row status: active

teldatgroup                       SecName: teldatuser
                                  Security model: v3 (USM)
                                  Storage Type: permanent
                                  Row status: active
```

3.2.2.6 LIST HOST

Lists information on the configured notifications.

Syntax:

```
SNMP+list host
```

Example:

```
SNMP+list host
          Host Address              Parameters
-------------------------------   -------------------------
172.24.51.12                      UDP port: 162
                                  Notification type: trap
                                  SecName: private
                                  Message Processing Model: v1
                                  security: v1 noAuthNoPriv
```

3.2.2.7 LIST STATISTICS

Lists statistics on packets that are sent and received.

Syntax:

```
SNMP+list statistics
```

Example:

```
SNMP+list statistics
----------------------------------------
              SNMP Counters
----------------------------------------
In Packets ..................... 0
        In Bad Versions ............. 0
        In Bad Community Names ...... 0
        In Bad Community Uses ....... 0
        In ASN Parse Errors ......... 0
        In Total Request Variables .... 0
        In Total Set Variables ...... 0
        In GET Requests ............. 0
        In GET-NEXT Requests ........ 0
        In SET Requests ............. 0
        In GET-RESPONSEs ............ 0
        In Traps .................... 0
Out Packets .................... 0
        Out Packets Too Big ......... 0
        Out No Such Names ........... 0
        Out Bad Values .............. 0
        Out Generic Errors .......... 0
        Out GET-RESPONSEs ........... 0
        Out Traps ................... 0
```

```
----------------------------------------
             USM Counters
----------------------------------------
Unsupported Security Levels ..... 0
Not In Time Windows ............. 0
Unknown User Names .............. 0
Unknown Engine IDs .............. 0
Wrong Digests ................... 0
Description Errors .............. 0

----------------------------------------
           Target MIB Counters
----------------------------------------
Unknown Contexts ................ 0

----------------------------------------
             Other Counters
----------------------------------------
Unknown Security Models ......... 0
Invalid Messages ................ 0
Unknown PDU Handlers ............ 0
SNMP+
```

3.2.2.8 LIST TRAP-SENDING-PARAMETERS

Lists information related to trap-sending.

Syntax:

```
SNMP+list trap-sending-parameters
```

Example:

```
SNMP+list trap-sending-parameters
Max time storing traps (seg):                    50
Max number of traps to store:                    32
Manager reachability check before sending traps: UDP echo

SNMP+
```

3.2.2.9 LIST USER

Lists information about the configured users.

Syntax:

```
SNMP+list user
```

Example:

```
SNMP+list user
           User Name                Parameters
---------------------------------   --------------------------
teldatuser
                                    Engine ID: 1234567890
                                    Group-name: teldatgroup
                                    SecName: teldat
                                    Authentication Protocol: SHA
                                    Privacy Protocol: AES128
```

3.2.2.10 LIST VIEW

Lists information on the views defined in the system.

Syntax:

```
SNMP+list view
```

Example:

```
SNMP+list view
_all_
                                  view subtree: .0
                                  view mask:
                                  view type: included
                                  Storage Type: permanent
                                  Row status: active

_all_
                                  view subtree: .1
                                  view mask:
                                  view type: included
                                  Storage Type: permanent
                                  Row status: active

_all_
                                  view subtree: .2
                                  view mask:
                                  view type: included
                                  Storage Type: permanent
                                  Row status: active

_none_
                                  view subtree: .0
                                  view mask:
                                  view type: excluded
                                  Storage Type: permanent
                                  Row status: active

_none_
                                  view subtree: .1
                                  view mask:
                                  view type: excluded
                                  Storage Type: permanent
                                  Row status: active

_none_
                                  view subtree: .2
                                  view mask:
                                  view type: excluded
                                  Storage Type: permanent
                                  Row status: active
SNMP+
```

3.2.3 EXIT

Exits the SNMP monitoring menu.

Syntax:

```
SNMP+exit
```

Example:

```
SNMP+exit
+
```

Chapter 4 Configuration Examples

4.1 SNMPv1

Configuring a *private* community with read and write access, accessible from subnet 172.24.0.0.

```
      community private access write-read-trap
      community private subnet 172.24.0.0 255.255.0.0
```

4.2 SNMPv3

Configuring four different users applying the different possible authentication and encryption combinations. Said users have access to all MIBs.

We also configure the engineID to simplify data collection and decoding when using a traffic analyst.

First configure the engineID:

```
SNMP config>engineid local 1234567890
SNMP config>
```

Next, configure the four different users:

```
SNMP config>user teldatshaaes auth sha plain teldatshakey1 priv aes128 plain teldataeskey1
SNMP config>user teldatmd5aes auth md5 plain teldatmd5key1 priv aes128 plain teldataeskey2
SNMP config>user teldatmd5des auth md5 plain teldatmd5key2 priv des plain teldatdeskey1
SNMP config>user teldatshades auth sha plain teldatshakey2 priv des plain teldatdeskey2
```

Assign the users to a group:

```
SNMP config>group teldatshaaes v3-usm teldatgroup
SNMP config>group teldatmd5aes v3-usm teldatgroup
SNMP config>group teldatmd5des v3-usm teldatgroup
SNMP config>group teldatshades v3-usm teldatgroup
```

Finally, define the views associated with said group. Use the _all_ view to indicate all OIDs.

```
SNMP config>access teldatgroup v3-usm priv read-view _all_ write-view _all_ noti
fy-view _all_
```

The final configuration is as follows:

```
SNMP config>sho conf
; Showing Menu and Submenus Configuration for access-level 15 ...
; Super Router * * Version 10.8.0-Alfa

      engineid local 1234567890
      user teldatmd5aes auth md5 ciphered 0x9EF49A1E6DE98B3A17B395CC91972DD1 priv
 aes128 ciphered 0xF85A722D9DFE44BD02B19BD7FDF39A31
      user teldatmd5des auth md5 ciphered 0x9EF49A1E6DE98B3A3B301FEBBE355690 priv
 des ciphered 0xC5577087CC1FC12A979CDB77D6EE1682
      user teldatshaaes auth sha ciphered 0xABF7853ECFB670CAF5830731ECEF2D4F priv
 aes128 ciphered 0xF85A722D9DFE44BD56ADCA608C25C3EE
      user teldatshades auth sha ciphered 0xABF7853ECFB670CA26C2B4252C567511 priv
 des ciphered 0xC5577087CC1FC12AB10CF76833CD4F80
;
      group teldatmd5aes v3-usm teldatgroup
;
      group teldatmd5des v3-usm teldatgroup
;
      group teldatshaaes v3-usm teldatgroup
;
      group teldatshades v3-usm teldatgroup
;
;
      access teldatgroup v3-usm priv read-view _all_ write-view _all_ notify-view _all_
;
```

```
;
SNMP config>
```

4.3 SNMPv1/v2 Multi-VRF Consults

Configuring a context and associating it with a secondary VRF that you wish to consult.

Define this context in the SNMP protocol menu and associate it with the relevant community.

```
   log-command-errors
   no configuration
   feature vrf
; -- VRF user configuration --
      vrf vrf1 context ctxt
   exit
;
   network ethernet0/0
; -- Ethernet Interface User Configuration --
      ip address 192.168.10.1 255.255.255.0
;
   exit
;
;
   protocol snmp
; -- SNMP user configuration --
      community second context ctxt
;
      context ctxt
;
;
   exit
;
   dump-command-errors
   end
```

4.4 SNMPv3 Multi-VRF Consults

Configuring a context and associating it with a secondary VRF that you wish to consult.

Define this context in the SNMP protocol menu and associate it with the **access** command.

```
   log-command-errors
   no configuration
   feature vrf
; -- VRF user configuration --
      vrf vrf1 context ctxt
   exit
;
   network ethernet0/0
; -- Ethernet Interface User Configuration --
      ip address 192.168.10.1 255.255.255.0
;
   exit
;
;
   protocol snmp
; -- SNMP user configuration --
      user snmp_user
;
      group snmp_user v3-usm snmp_group
;
      access snmp_group context ctxt v3-usm noauth
;
      context ctxt
;
;
```

```
   exit
;
   dump-command-errors
   end
```