

ComSNMP 1.0 Trial Version Programmer's Guide©

Rig Consulting, Inc.

<http://www.snmp-component.com>

<http://www.rigconsulting.com>

ComSNMP 1.0 Trial Version Programmer's Guide

Rig Consulting, Inc.

<http://www.snmp-component.com>

<http://www.rigconsulting.com>

Table of Contents

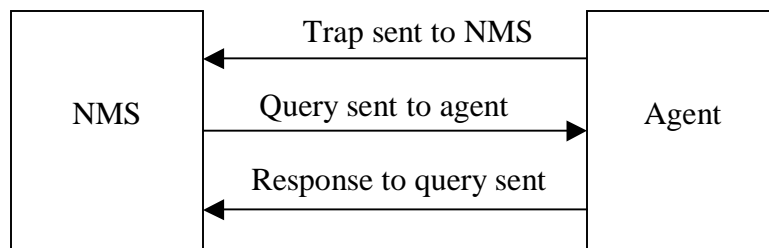
ComSNMP 1.0 Trial Version Programmer's Guide	1
1 Overview	3
2 ComSNMP	3
3 ComSNMP COM Requirements for Client Programs	4
3.1 Static Library Requirements for Client Programs.....	4
3.2 Component Registration	5
3.3 Component Distribution	5
4 SNMP Interface	5
4.1 GetSingle	5
4.2 GetMultiple.....	6
4.3 GetBulk.....	7
4.4 GetNext.....	8
4.5 InitMIBWalk.....	8
4.6 MIBWalk	8
4.7 GetRecord.....	8
4.8 ResetMIBWalk	8
4.9 SetSingle	8
4.10 SetMultiple	9
4.11 SetVersion.....	9
4.12 SetCommunity	9
4.13 GetCommunity.....	9
4.14 SetAgentIP.....	9
4.15 ConfigureReliability.....	10
4.16 SetUSMConfig.....	10
4.17 GetUSMConfig	10
4.18 GetErrorMessage	11
5 Soliciting and Receiving Traps	12
5.1 Receiving Traps	12
5.2 Soliciting Traps Using ComSNMP Interface	12
5.2.1 ConfigureTrap	12
5.2.2 SetTrapForwarding.....	12
6 Static Library Interface	12
6.1 InitDefaults().....	13
6.2 ResetDefaults().....	13
7 Guidelines for using Set Functions from VB Sample Client	13
8 Error Codes.....	14
8.1 ComSNMP specific HRESULT Error Codes.....	14
8.2 Net-snmp specific Error Codes and Messages	16
9 Traps	19
9.1 Notes on Trap and ConfigureTrap	20
9.2 SetTrapForwarding	20

1 Overview

SNMP (Simple Network Management Protocol) was introduced to meet the growing need for a standard for managing Internet Protocol (IP) devices. SNMP provides its users with a simple set of operations that allows these devices to be managed remotely.

The core of SNMP is a simple set of operations that gives administrators the ability to change the state of some SNMP-based device. SNMP is usually associated with managing routers, printers, modem racks, power supplies and more. Any device running software that allows the retrieval of SNMP information can be managed. This includes not only physical devices but also software, such as web servers and databases. Network monitoring can also be done with the help of SNMP.

In the world of SNMP there are two kinds of entities: managers and agents. A manager is a server running some kind of software system that can handle management tasks for a network. Managers are often referred to as Network Management Stations (NMS). The agent is a piece of software that runs on the network devices you are managing.



2 ComSNMP

ComSNMP is developed using C++. On Win32 platforms ComSNMP can be either used as COM (Component Object Model) component or Static Library. Programmers can use ComSNMP to develop NMS applications requiring SNMP V1, V2c or V3. ComSNMP code is based on the net-snmp (version 5.0.8) open source code.

Features of ComSNMP :

- ComSNMP is tri-lingual and supports v1, v2c and v3 (RFC 2576).
- Strict compatibility with SNMP V3 RFCs.
- Interface for get, get multiple using single PDU, bulk-get, get-next, set, set multiple using single PDU, trap (send) and trap (receive) for SNMP V3 and whatever subset is applicable to V1 and V2c.
- Choices to forward traps (Notifications) to a remote host, forwarding traps to the NMS application, ignoring traps and soliciting trap to a custom port.
- Ability to receive trap information in Binary and Record formats.

- Supports USM (User Based Security Model). HMAC-MD5-96(authentication) and CBC-DES (security/encryption) protocols.
- User-friendly interfaces to walk tables and MIB's in one method call given the dotted notation prefix for the MIB.
- Support for all Windows 32 bit platforms starting from Windows NT 4.0 onwards.
- Components accessible from scripting and compiled languages.
- Self-registering components.
- Will cut your NMS development time by multifold – if you have written original SNMP V1 code or customized open source code for your NMS application, you know what we are talking about!.
- ComSNMP is not MIB aware and works on Objects by their Object IDs – hence your existing NMS applications can continue to use the existing scheme for reading MIBs. This makes ComSNMP a generic framework, which you use with existing NMS and related tools to support V3.
- Extensive VC++ and VB standalone and trilingual sample client programs using ComSNMP along with source code included with the full-feature product.

3 ComSNMP COM Requirements for Client Programs

- ProjSNMP.dll

This is a COM inproc server, which contains the implementation for the IDL interface. The ProjSNMP.dll will not be registered in any system unless the Libeay32.dll is copied to <WINDOWS>\System32.

- libeay32.dll

This is the OpenSSL binary (DLL) for supporting DES and SHA encryption algorithms for SNMP v3. The ProjSNMP.dll will not be registered in any system unless the libeay32.dll is copied to winnt\system32.

3.1 Static Library Requirements for Client Programs

- YS_snmplib.lib

ComSNMP static library for Win32 Platforms.

- linsnmp.lib

Open source code library for SNMP.

- libeay32.lib

OpenSSL library for supporting DES and SHA encryption algorithms for SNMP v3. This library is statically linked with YS_snmplib.lib.

3.2 Component Registration

- Copy ProjSNMP.dll and Libeay32.dll to <WINDOWS>\System32 directory. ComSNMP registration pre-requires libeay32.dll copied to <WINDOWS>\System32 directory.
- Make sure TCP/IP protocol is enabled on your system. Most systems have this enabled. If you are in doubt check with your systems administrator.
- Register the license key using the Install.exe program from command line:
>Install <LICENCE_KEY>
This step is not required for Trial versions of the product.
- Register the component using the command
>regsvr32 ProjSNMP.dll
Click OK on the dialog box displayed that confirms registration.

3.3 Component Distribution

While distributing your product developed using ComSNMP, you will have to include the following components to be copied to the <WINDOWS>\System32 directory and make a call to the Install.exe program by passing the license key as command line argument, from your end product's installation program.

- libeay32.dll
- ProjSNMP.dll

Trial version of ComSNMP can't be distributed and hence the above does not apply to Trial versions of the product.

4 SNMP Interface

This section documents the SNMP Interface provided by ComSNMP which is to be used by the application you will develop using ComSNMP i.e, your NMS. The COM component version implements the interface via proprietary IComSNMP.

4.1 GetSingle

Method	HRESULT GetSingle ([in] BSTR bstrOID, [out] VARIANT *varReturnedValue, [out] long *nSnmpErrCode);
Arguments	[in] BSTR bstrOID : Object ID whose value needs to be fetched from the Agent [out] VARIANT *varReturnedValue:

	HRESULT = S_OK
Comments	In SNMP V1, bulk-get operation is not supported using which more number of OID values can be fetched from the Agent at one shot using single PDU. This method is for the benefit of V1 wherein bulk-get is not supported.

4.3 GetBulk

Method	HRESULT GetBulk ([in] SAFEARRAY(BSTR) *sarrOIDs, [out] SAFEARRAY(VARIANT) *varReturnedValues, [out] SAFEARRAY(long) *nsarrSnmprErrCode, [out] long *nSnmprErrCode);
Arguments	<p>[in] SAFEARRAY(BSTR) *sarrOIDs : Object IDs whose values need to be fetched from the Agent</p> <p>[out] SAFEARRAY(VARIANT) *varReturnedValues : Values returned from the Agent if the method executed successfully.</p> <p>[out] SAFEARRAY(long) *nsarrSnmprErrCode : Error codes corresponding to the OIDs sent to the agent; 0 indicates the OID is returned successfully; value other than 0 indicates the SNMP error returned by the agent</p> <p>[out] long *nSnmprErrCode : if the method executed successfully this value is 0; values other than 0 indicate the SNMP errors as provided by the agent</p>
Return Value	HRESULT = S_OK & nSnmprErrCode = 0 in case of success. Lookup HRESULT for COM specific error and nSnmprErrCode for SNMP specific errors in case of failure.
Example	<p>To get the values in bulk from system.sysDescr of MIB2 : (provided) :</p> <pre>sarrOIDs [0] = "1.3.6.1.2.1.1.1.0" GetBulk(&sarrOIDs,&varReturnedValues,&sarrSnmprErrCode,&nSnmprErrCode);</pre> <p>(returned) :</p> <pre>iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.8072.3.2.255 iso.3.6.1.2.1.1.3.0 = Timeticks: (6712) 0:01:07.12 iso.3.6.1.2.1.1.4.0 = STRING: "admin@ys.com" iso.3.6.1.2.1.1.5.0 = STRING: "YS Router 2100" iso.3.6.1.2.1.1.6.0 = STRING: "telephone closet, 3rd floor" iso.3.6.1.2.1.1.7.0 = INTEGER: 72 iso.3.6.1.2.1.1.8.0 = Timeticks: (14) 0:00:00.14 iso.3.6.1.2.1.1.9.1.2.1 = OID: iso.3.6.1.2.1.31 iso.3.6.1.2.1.1.9.1.2.2 = OID: iso.3.6.1.2.1.49 iso.3.6.1.2.1.1.9.1.2.3 = OID: iso.3.6.1.2.1.4 nSnmprErrCode = 0 (No Error) HRESULT = S_OK</pre>
Comments	By default, the next 10 OIDs of the given OID will be fetched; OIDs of different MIB's can also be given as the input.

4.4 GetNext

Method	HRESULT GetNext ([in] BSTR bstrOID, [out] BSTR *bstrOIDout, [out] long *nSnmprErrCode);
Arguments	[in] BSTR bstrOID : Object ID whose next OID needs to be fetched from the Agent [out] BSTR *bstrOIDout : The next OID of the given OID in the MIB. [out] long *nSnmprErrCode : nSnmprErrCode = 0 (No Error) SNMP error code as provided by the Agent
Return Value	HRESULT = S_OK & nSnmprErrCode = 0 in case of success. Lookup HRESULT for COM specific error and nSnmprErrCode for SNMP specific errors in case of failure.
Example	To get the next OID of MIB2 system.sysDescr: (provided) bstrOID = "1.3.6.1.2.1.1.1.0" GetNext(bstrOid,&bstrOidOut,&nSnmprErrCode); (returned) bstrOIDout = 1.3.6.1.2.1.1.2.0 nSnmprErrCode = 0 (No Error) HRESULT = S_OK

4.5 InitMIBWalk

Not available in Trial Version.

4.6 MIBWalk

Not available in Trial Version.

4.7 GetRecord

Not available in Trial Version.

4.8 ResetMIBWalk

Not available in Trial Version.

4.9 SetSingle

Not available in Trial Version.

4.10 SetMultiple

Not available in Trial Version.

4.11 SetVersion

Method	HRESULT SetVersion ([in] long nSnmVersion, [out] long *nSnmErrCode);
Arguments	[in] long nSnmVersion : SNMP Version to be set [out] long *nSnmErrCode) : SNMP error code as provided by agent
Return Value	HRESULT = S_OK & nSnmErrCode = 0 in case of success. Lookup HRESULT for COM specific error and nSnmErrCode for SNMP specific errors in case of failure.
Example	To set the version as SNMP version 1 : (provided) nSnmVersion = 0 SetVersion(nSnmVersion,&nSnmErrCode); (returned) nSnmErrCode = 0 (No Error) HRESULT = S_OK
Comments	The following are the values corresponding to the SNMP versions as used in net-snmp : SNMP V1 - 0 SNMP V2c - 1 SNMP V3 - 3

4.12 SetCommunity

Not available in Trial Version.

4.13 GetCommunity

Not available in Trial Version.

4.14 SetAgentIP

Method	HRESULT SetAgentIP ([in] BSTR bstrAgentIP, [in] long nPort, [out] long *nSnmErrCode);
Arguments	[in] BSTR bstrAgentIP : IP address of the agent to be set [in] long nPort :

	Port number to be set on which the agent is listening for requests [out] long *nSnmpErrCode) : SNMP error code as provided by agent
Return Value	HRESULT = S_OK & nSnmpErrCode = 0 in case of success. Lookup HRESULT for COM specific error and nSnmpErrCode for SNMP specific errors in case of failure.
Example	To set the agentip = "10.0.2.20" and port no = 5000 (provided) bstrAgentIP = "10.0.2.20" nPort = 5000 SetAgentIP(bstrAgentIP, nPort, nSnmpErrCode); (returned) nSnmpErrCode = 0 (No Error) HRESULT = S_OK

4.15 ConfigureReliability

Not available in Trial Version.

4.16 SetUSMConfig

Not available in Trial Version.

The Trial Version uses a default configuration provided in the SNMP V3 sample test configuration file ""sample_v3_config.txt"

Please configure your agent per this configuration to get a feel of SNMP V3 features.

4.17 GetUSMConfig

Method	HRESULT GetUSMConfig ([out] BSTR *bstrUserName, [out] long *nSnmpSecurityLevel, [out] BSTR *bstrAuth, [out] BSTR *bstrSec, [out] BSTR *bstrPassword, [out] BSTR *bstrEncryptPassword, [out] long *bSecPassword, [out] long *nReserved, [out] long *nSnmpErrCode);
Arguments	[out] BSTR bstrUserName : UserName to get [out] long nSnmpSecurityLevel : SNMP Security Level(noauthnopriv/authnopriv/authpriv) to get [out] BSTR bstrAuth : Authentication type (MD5/SHA) to get [out] BSTR bstrSec : Privacy type (DES) to get [out] BSTR bstrPassword : Authentication Password [out] BSTR bstrEncryptPassword : Privacy Password

	<p>[out] long bSecPassword : To get the status(enable/disable) of encryption</p> <p>[out] long nReserved : Reserved for future use</p> <p>[out] long *nSnmprErrCode) : SNMP error code as provided by agent</p>
Return Value	<p>HRESULT = S_OK & nSnmprErrCode = 0 in case of success. Lookup HRESULT for COM specific error and nSnmprErrCode for SNMP specific errors in case of failure.</p>
Example	<p>To get the value of MIB2 system.sysDescr: (provided):</p> <pre>GetUSMConfig(&bstrUserName, &nSnmprSecurityLevel, &bstrAuth, &bstrSec, &bstrPassword, &bstrEncryptPassword, &bSecPassword, &nReserved, &nSnmprErrCode);</pre> <p>(returned)</p> <pre>bstrUserName : myname nSnmprSecurityLevel : 3 (authpriv) bstrAuth : SHA bstrSec : DES bstrPassword : myshapasswd bstrEncryptPassword : mydespasswd bSecPassword : 1 nReserved : 0 nSnmprErrCode = 0 (No Error) HRESULT = S_OK</pre>

4.18 GetErrorMessage

Method	HRESULT GetErrorMessage ([in] long nErrCode, [out] BSTR *bstrErrMesg);
Arguments	<p>[in] long nErrCode :</p> <p>[out] BSTR *bstrErrMesg :</p>
Return Value	<p>HRESULT = S_OK & nSnmprErrCode = 0 in case of success. Lookup HRESULT for COM specific error and nSnmprErrCode for SNMP specific errors in case of failure.</p>
Example	<p>To get the error message of the errcode -58 :</p> <p>(provided) :</p> <pre>nErrCode = -58 GetErrorMessage(nErrCode,&bstrErrMesg);</pre> <p>(returned) :</p> <pre>bstrErrMesg = BSTR/"Unknown Object Identifier" nSnmprErrCode = 0 (No Error) HRESULT = S_OK</pre>
Comments	Please lookup sec.8 for the complete list of error codes and explanation.

5 Soliciting and Receiving Traps

This section documents the SNMP Trap Interface for receiving Traps and SNMP Interface methods to solicit traps from the SNMP Agent.

5.1 Receiving Traps

The SNMP Trap Interface definition is provided by ComSNMP which is to be **implemented by the application you develop using ComSNMP i.e, your NMS**, in order to receive traps and notifications received and forwarded by ComSNMP DLL to your NMS program.

The IComSNMPTrapSink interface is used to receive traps and has only one method by name "Trap" and ComSNMP has not implemented this method. Because the client has to implement this function and listen on IComSNMPTrapSink interface for incoming traps. It is up to the client implementation to decide what to do with the received traps.

Please refer to the sample source of "**TrapClient**" for the implementation of Trap. Our sample just receives the trap message and displays it on the screen. Please refer to **sec.9** for more details on Trap.

"TrapClient" is not provided with the Trial Version.

5.2 Soliciting Traps Using ComSNMP Interface

5.2.1 ConfigureTrap

Not available in Trial Version.

5.2.2 SetTrapForwarding

Not available in Trial Version.

6 Static Library Interface

The static library interface resembles the COM interface, the only difference being when you link this library with your application (and make calls to the library) the COM plumbing is not used. This choice is provided for customers who desire performance. The static library interface file is ys-staticlib-utils.h

The following helper functions have been added for the ease of configuration of the ComSNMP static library. Since ComSNMP uses default values for Community string, Agent IP address, Version

and other parameters, these functions have been introduced to initialize and reset the default values when the static library interface is used.

6.1 InitDefaults()

Method	Void InitDefaults() ;
Arguments	Nil
Return Value	In case of success 0 is returned If case of failure non-zero values are returned
Example	To initialize default values for Static Library: (provided) InitDefaults(); (returned) 0 (success)
Comments	Before calling Static Library functions, this function must be invoked to initialize default values.

6.2 ResetDefaults()

Method	Void ResetDefaults() ;
Arguments	Nil
Return Value	In case of success 0 is returned If case of failure non-zero values are returned
Example	To reset default values: (provided) ResetDefaults(); (returned) 0 (success)
Comments	After calling Static Library functions, this function must be invoked to reset the default values.

7 Guidelines for using Set Functions from VB Sample Client

For set functions, the syntax for the input while using the VB Sample Client program window should follow this format:

OID type value

Example:

To set the OID 1.3.6.1.2.1.1.4.0 whose type is STRING, follow the syntax given below :

1.3.6.1.2.1.1.4.0 s admin@ys.com

Similarly, data types corresponding to various kinds of OIDS are given below:

OID	TYPE
STRING, OID, TimeTicks, Hex-String, IP Address, Octet String, BITS	S
signed integer (both 2 byte & 4 byte)	I
unsigned integer(both 2 byte & 4 byte), gauge32, counter32	u
signed long integer	I
unsigned long integer (8 byte)	U
float (4 byte)	F
double (8 byte)	D

8 Error Codes

8.1 ComSNMP specific HRESULT Error Codes

HRESULT : E_COMSNMP_MALLOCCERR (0x8004FF00L)

Message : "Memory allocation failure"

- This error is returned in case of memory allocation failure.

HRESULT : E_COMSNMP_AGENTIP_NULL (0x8004FF01L)

Message : "AgentIP is null",

- While configuring the AgentIP using the function SetAgentIP, if the value of input AgentIP is NULL then this error is returned. Give the value for AgentIP and call SetAgentIP again.

HRESULT : E_COMSNMP_COMMUNITY_NULL (0x8004FF02L)

Message : "Community string is null"

- While configuring the community string using the function SetCommunity, if the value of input community string is NULL then this error is returned. Give the value for Community string and call SetCommunity again.

HRESULT : E_COMSNMP_BAD_VERSION (0x8004FF03L)

Message : "Bad SNMP version"

- While configuring the SNMP version using the function SetVersion, if the value of input version is not valid then this error is returned. Give the value for version and call SetVersion again.

HRESULT : E_COMSNMP_INVALID_TIMEOUT (0x8004FF04L)

Message : "Invalid timeout value"

- While configuring the timeout value using the function ConfigReliability, if the value of input timeout is not valid (for example a negative value) then this error is returned. Give the valid value for timeout and call ConfigReliability again.

HRESULT : E_COMSNMP_INVALID_RETRIES (0x8004FF05L)

Message : "Invalid retries value"

- While configuring the retries value using the function ConfigReliability, if the value of input retries is not valid (for example a negative value) then this error is returned. Give the valid value for retries and call ConfigReliability again.

HRESULT : E_COMSNMP_NEED_USMCONFIG (0x8004FF06L)

Message : "Please use SetUSMConfig & configure USM details first"

- Without configuring the USM details if the user tries to use SNMP V3 then this error is returned. So, call SetUSMConfig with USM details first and continue SNMP operations.

HRESULT : E_COMSNMP_INVALID_SECLEVEL (0x8004FF07L)

Message : "Invalid security level"

- While configuring the USM details using SetUSMConfig, if the value of Security Level is NULL or invalid then this error is returned. Please use the following values for various SNMP security levels as used in net-snmp :

Noauthnopriv - 1

Authnopriv - 2

Authpriv - 3

HRESULT : E_COMSNMP_USERNAME_NULL (0x8004FF08L)

Message : "Username is null"

- While configuring the USM details using SetUSMConfig, if the value of UserName is NULL then this error is returned. Give the value for UserName and call SetUSMConfig again.

HRESULT : E_COMSNMP_AUTHTYPE_NULL (0x8004FF09L)

Message : "Authentication type is null"

- While configuring the USM details using SetUSMConfig, if the value of Authentication type is NULL then this error is returned. Give the value for authentication type and call SetUSMConfig again.

HRESULT : E_COMSNMP_AUTHPASSWD_NULL (0x8004FF0AL)

Message : "Authentication password is null"

- While configuring the USM details using SetUSMConfig, if the value of Authentication password is NULL then this error is returned. Give the value for authentication password and call SetUSMConfig again.

HRESULT : E_COMSNMP_PRIVACY_NULL (0x8004FF0BL)

Message : "Privacy type is null"

- While configuring the USM details using SetUSMConfig, if the value of Privacy(Encryption) type is NULL then this error is returned. Give the value for Privacy and call SetUSMConfig again.

HRESULT : E_COMSNMP_PRIVPASSWD_NULL (0x8004FF0CL)

Message : "Privacy password is null"

- While configuring the USM details using SetUSMConfig, if the value of Privacy(Encryption) password is NULL then this error is returned. Give the value for privacy password and call SetUSMConfig again.

HRESULT : E_COMSNMP_ENDOFMIB (0x8004FF0FL)

Message : "End of MIB"

- During MIBWalk, if the Agent finds that the traversal is complete and it has reached the end of MIB or the given OID is invalid then this error is returned.

8.2 Net-snmp specific Error Codes and Messages

The net-snmp specific error codes could be returned either from the client or the agent. That is, if there is any error when a request PDU is formed or it is validated, client returns respective error codes. Similarly, the agent will return errors in the response PDU as a result of the performed SNMP operation.

The following errors are returned in the response PDU from the SNMP Agent. These error codes have been taken from snmplib\snmp_client.c.

Error Code	Error Message
0	(noError) No Error

1	(tooBig) Response message would have been too large.
2	(noSuchName) There is no such variable name in this MIB.
3	(badValue) The value given has the wrong type or length.
4	(readOnly) The two parties used do not have access to use the specified SNMP PDU.
5	(genError) A general failure occurred
6	noAccess
7	wrongType (The set datatype does not match the data type the agent expects)
8	wrongLength (The set value has an illegal length from what the agent expects)
9	wrongEncoding
10	wrongValue (The set value is illegal or unsupported in some way)
11	noCreation (that table does not support row creation)
12	inconsistentValue (The set value is illegal or unsupported in some way)
13	resourceUnavailable (This is likely a out-of-memory failure within the agent)
14	commitFailed
15	undoFailed
16	authorizationError (access denied to that object)"
17	notWritable (that object does not support modification)
18	inconsistentName

The following errors are returned by ComSNMP when the agent's validation of the request PDU from NMS/ComSNMP fails. These error codes have been taken from `snmplib\snmp_api.c`.

Sl.No.	Error Code	Error message
1.	SNMPERR_SUCCESS	"No error"
2.	SNMPERR_GENERR	"Generic error"
3.	SNMPERR_BAD_LOCPORT	"Invalid local port"
4.	SNMPERR_BAD_ADDRESS	"Invalid Address"
5.	SNMPERR_BAD_ADDRESS	"Unknown host"
6.	SNMPERR_BAD_SESSION	"Unknown session"
7.	SNMPERR_TOO_LONG	"Too long"
8.	SNMPERR_NO_SOCKET	"No socket"

9.	SNMPERR_V2_IN_V1	"Cannot send V2 PDU on V1 session"
10.	SNMPERR_V1_IN_V2	"Cannot send V1 PDU on V2 session"
11.	SNMPERR_BAD_REPEATERS	"Bad value for non-repeaters"
12.	SNMPERR_BAD_REPETITION S	"Bad value for max-repetitions"
13.	SNMPERR_BAD_ASN1_BUILD	"Error building ASN.1 representation"
14.	SNMPERR_BAD_SENDTO	"Failure in sendto"
15.	SNMPERR_BAD_PARSE	"Bad parse of ASN.1 type"
16.	SNMPERR_BAD_VERSION	"Bad version specified"
17.	SNMPERR_BAD_SRC_PARTY	"Bad source party specified"
18.	SNMPERR_BAD_DST_PARTY	"Bad destination party specified"
19.	SNMPERR_BAD_CONTEXT	"Bad context specified"
20.	SNMPERR_BAD_COMMUNITY	"Bad community specified"
21.	SNMPERR_NOAUTH_DESPRIV	"Cannot send noAuth/Priv"
22.	SNMPERR_BAD_ACL	"Bad ACL definition"
23.	SNMPERR_BAD_PARTY	"Bad Party definition"
24.	SNMPERR_ABORT	"Session abort failure"
25.	SNMPERR_UNKNOWN_PDU	"Unknown PDU type"
26.	SNMPERR_TIMEOUT	"Timeout"
27.	SNMPERR_BAD_RECVFROM	"Failure in recvfrom"
28.	SNMPERR_BAD_ENG_ID	"Unable to determine contextEngineID"
29.	SNMPERR_BAD_SEC_NAME	"No securityName specified"
30.	SNMPERR_BAD_SEC_LEVEL	"Unable to determine securityLevel"
31.	SNMPERR_ASN_PARSE_ERR	"ASN.1 parse error in message"
32.	SNMPERR_UNKNOWN_SEC_ MODEL	"Unknown security model in me"
33.	SNMPERR_INVALID_MSG	"Invalid message (e.g. msgFlags)"
34.	SNMPERR_UNKNOWN_ENG_I D	"Unknown engine ID"
35.	SNMPERR_UNKNOWN_USER_ NAME	"Unknown user name"
36.	SNMPERR_UNSUPPORTED_SE C_LEVEL	"Unsupported security level"
37.	SNMPERR_AUTHENTICATION _FAILURE	"Authentication failure (incorrect password,community or key)"
38.	SNMPERR_NOT_IN_TIME_WI NDOW	"Not in time window"
39.	SNMPERR_DECRYPTION_ERR	"Decryption error"
40.	SNMPERR_SC_GENERAL_FAI LURE	"SCAPI general failure"
41.	SNMPERR_SC_NOT_CONFIGU	"SCAPI sub-system not configured"

	RED	
42.	SNMPERR_KT_NOT_AVAILABLE	"Key tools not available"
43.	SNMPERR_UNKNOWN_REPORT	"Unknown Report message"
44.	SNMPERR_USM_GENERICERROR	"USM generic error"
45.	SNMPERR_USM_UNKNOWNSECURITYNAME	"USM unknown security name (no such user exists)"
46.	SNMPERR_USM_UNSUPPORTEDSECURITYLEVEL	"USM unsupported security level of security"
47.	SNMPERR_USM_ENCRYPTIONERROR	"USM encryption error"
48.	SNMPERR_USM_AUTHENTICATIONFAILURE	"USM authentication failure (incorrect password or key)"
49.	SNMPERR_USM_PARSEERROR	"USM parse error"
50.	SNMPERR_USM_UNKNOWNENGINEID	"USM unknown engineID"
51.	SNMPERR_USM_NOTINTIMEWINDOW	"USM not in time window"
52.	SNMPERR_USM_DECRYPTIONERROR	"USM decryption error"
53.	SNMPERR_NOMIB	"MIB not initialized"
54.	SNMPERR_RANGE	"Value out of range"
55.	SNMPERR_MAX_SUBID	"Sub-id out of range"
56.	SNMPERR_BAD_SUBID	"Bad sub-id in object identifier"
57.	SNMPERR_LONG_OID	"Object identifier too long"
58.	SNMPERR_BAD_NAME	"Bad value name"
59.	SNMPERR_VALUE	"Bad value notation"
60.	SNMPERR_UNKNOWN_OBJID	"Unknown Object Identifier"
61.	SNMPERR_NULL_PDU	"No PDU in snmp_send"
62.	SNMPERR_NO_VARS	"Missing variables in PDU"
63.	SNMPERR_VAR_TYPE	"Bad variable type"
64.	SNMPERR_MALLOC	"Out of memory (malloc failure)"
65.	SNMPERR_KRB5	"Kerberos related error"

9 Traps

A trap is a way for an agent to tell the NMS that an interesting event has occurred. Traps provide a way for an agent to send a monitoring station asynchronous notification about conditions that the NMS should know about. The traps that an agent can generate are defined by the MIBs it supports; the number of traps can range from zero to hundreds. To see what traps are defined in any

MIB file, search the term "TRAP-TYPE" (SMIv1) or "NOTIFICATION-TYPE" (SMIv2) in the MIB file.

Traps fall into two categories, generic and enterprise-specific. There are seven generic trap numbers (0-6), such as coldstart(0), warmstart(1), linkDown(2), linkup(3), authenticationFailure(4), egpNeighborLoss(5) and enterpriseSpecific(6).

An Enterprise-specific trap is identified by two pieces of information: the enterprise ID of the organization that defined the trap and a specific trap number assigned by that organization. The notion of an enterprise-specific trap is extremely flexible, because organizations are allowed to subdivide their enterprises as much as they like. For example, if an enterprise number is 2789, the enterprise MIB OID prefix would be 1.3.6.1.4.1.2789. This can further be subdivided, defining traps with enterprise IDs such as 1.3.6.1.4.1.2789.5000, 1.3.6.1.4.1.2789.5001 and so on.

Handling incoming traps is the responsibility of the NMS. Some NMS programs do as little as display the incoming traps to standard output(stdout). However, an NMS server typically has the ability to react to the SNMP traps it receives. For example, when NMS receives a linkDown trap from a router, it might respond to the event by paging the contact person, displaying a pop-message on a management console, or forwarding even to another NMS.

9.1 Notes on Trap and ConfigureTrap

Not available in Trial Version.

9.2 SetTrapForwarding

Not available in Trial Version.