SNMP ASN.1 protocol module for TTCN-3 Toolset with TITAN, Function Description

Endre Kulcsár

Version 1551-CNL 113 774, Rev. A, 2013-03-29

Table of Contents

Functionality . . . 1
- Protocol Version Implemented . . . 1
- Modifications/Deviations Related to the Protocol Specification . . . 1
  - Unimplemented Messages, Information Elements and Constants . . . 1
  - Protocol Modifications/Deviations . . . 1
- System Requirements . . . 1

Usage . . . 1
- Installation . . . 2
- Examples . . . 2

Interface Description . . . 2
- Top Level PDU . . . 2
- Encoder/Decoder Functions . . . 2
  - Implemented Encoding/Decoding Functions: . . . 2

Terminology . . . 4

References . . . 4

Functionality

Protocol modules implement the message structures of the related protocol in a formalized way, using the standard specification language ASN.1 [7]. This allows defining of test data (templates) in the TTCN-3 language [1] and correctly encoding/decoding messages when executing test suites using the Titan TTCN-3 test environment.

Protocol Version Implemented

This set of protocol modules implements protocol messages and constants of the SNMP protocol, (see [2], [3], [4], [5]). SNMPv1, SNMPv2 and SNMPv3 messages are supported.

Modifications/Deviations Related to the Protocol Specification

None.

Unimplemented Messages, Information Elements and Constants

None.

Protocol Modifications/Deviations

None.

System Requirements

Protocol modules are a set of ASN.1 source code files that can be used as part of TTCN-3 test suites only. Hence, protocol modules alone do not put specific requirements on the system used. However, in order to compile and execute a TTCN-3 test suite using the set of protocol modules the following system requirements must be satisfied:

- TITAN TTCN-3 Test Executor version R7A (1.7.pl0) or higher installed.

NOTE This version of the protocol module is not compatible with TITAN releases earlier than R7A.

Usage

Installation

The set of protocol modules can be used in developing TTCN-3 test suites using any text editor. However to make the work more efficient a TTCN-3-enabled text editor is recommended (for example, `nedit`, `xemacs`). Since the SNMP protocol is used as a part of a TTCN-3 test suite, this requires TTCN-3 Test Executor be installed before the module can be compiled and executed together with other parts of the test suite. For more details on the installation of TTCN-3 Test Executor see the relevant section of [6].

Examples

The demo directory contains the file *SNMP_IPL4_Definitions.ttcn* which defines a dual face port. It can be used with the IPL4 test port to send and receive `SNMPv1_Message`, `SNMPv2_Message` or `SNMPv3_Message` in the test case in the same way it was implemented in the SNMP test port. The only difference is the dynamic update of the remote address. This can be performed now with the IPL4 test port's `f_IPL4_Connect` function.

The demo directory also contains the file *SNMP_Test.ttcn*. This `file` includes two test cases. The first test case is called `tc_SNMP_1` which sends and receives an `SNMPv1_Message`, an SNMPv2_Message and an SNMPv3_Message using the dual face port. The second test case is called `tc_SNMP_2`. It performs the same actions as the first test case, then the remote address is changed, then the actions of the first test case are repeated using the new remote address.

Interface Description

Top Level PDU

The top level PDUs are SNMPv1_Message, SNMPv2_Message and SNMPv3_Message.

Encoder/Decoder Functions

This product also contains encoding/decoding functions, which assure correct BER [8] encoding of messages when sent from Titan and correct BER [8] decoding of messages when received by Titan.

Implemented Encoding/Decoding Functions:

The following functions can be used for encoding and decoding of SNMP v1, v2 or v3 messages :

Name	Type of formal parameters	Type of return value
`enc_SNMPv1_Message`	in SNMPv1_Message	octetstring
`dec_SNMPv1_Message`	in octetstring	SNMPv1_Message

Name	Type of formal parameters	Type of return value
dec_SNMPv1_Message_backtrack	in octetstring, out SNMPv1_Message	integer (0: success, 1: decoding failed)
enc_SNMPv2_Message	in SNMPv2_Message	octetstring
dec_SNMPv2_Message	in octetstring	SNMPv2_Message
dec_SNMPv2_Message_backtrack	in octetstring, out SNMPv2_Message	integer (0: success, 1: decoding failed)
enc_SNMPv3_Message	in SNMPv3_Message	octetstring
dec_SNMPv3_Message	in octetstring	SNMPv3_Message
dec_SNMPv3_Message_backtrack	in octetstring, out SNMPv3_Message	integer (0: success, 1: decoding failed)

The following functions can be used for object identifier operations:

Name	Type of formal parameters	Type of return value	Functionality
oi_concat	in objid, in objid	objid	return concatenation of inputs
oi_size	in objid	integer	return number of elements of input
oi_eq	in objid, in objid	boolean	return true if two inputs are equal
oi_sub	in integer, in integer , in objid	objid	return sub-objid where first input is begin and second input is size
oi_gt	in objid, in objid	boolean	return true if first input > second input lexicographically
oi_addObjId	in objid, in SetOfInteger	Objid	return concatenation of first input with integers of second input
oi_setvalue	in objid, in integer, in integer	objid	insert into first input at position specified by the second input a value given by the third input
oi_getvalue	in objid input, in integer position	integer	return element selected by second input

The following functions can be used for authentication and encryption of SNMP v1, v2 or v3 messages :

Name	Type of formal parameters	Type of return value
calculate_MD5_MAC	in octetstring (authentication key),in SNMPv3_Message (message)	octetstring
encryptData	in octetstring (encryption key),in ScopedPDU (data to encrypt,out octetstring (encrypted data), in octetstring (privacy parameters)	-

Name	Type of formal parameters	Type of return value
`decryptData`	in octetstring (decryption key),in octetstring (privacy parameters),in octetstring (encrypted data), out ScopedPDU (decrypted data)	-
`password_to_key_md5`	in octetstring (password),in octetstring (engine ID),out octetstring (key)	-
`enc_UsmSecurityParameters`	in UsmSecurityParameters	octetstring
`dec_UsmSecurityParameters`	in octetstring	UsmSecurityParameters

Terminology

ASN.1

Abstract Syntax Notation One

ASP

Abstract Service Primitive

BER

Basic Encoding Rules

SNMP

Simple Network Management Protocol

TTCN-3

Testing and Test Control Notation version 3

References

[1] ETSI ES 201 873-1 v4.5.1
The Testing and Test Control Notation version 3. Part 1: Core Language

[2] RFC 1157
A Simple Network Management Protocol (SNMP)

[3] RFC 1901
Introduction to Community-based SNMPv2

[4] RFC 3411
Architecture for SNMP management frameworks

[5] RFC 3414
User-based Security Model (USM) for version 3 of the Simple NetworkManagement Protocol (SNMPv3)

[6] Programmer’s Technical Reference for TITAN TTCN–3 Test Executor

[7] ITU-T X.680 (07-2002)
Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation

[8] ITU-T X.690 (07-2002)
Information technology – ASN.1 encoding rules: Specification of basic encoding Rules (BER),Canonical encoding rules (CER) andDistinguished encoding rules (DER)