IPv6 READY Logo

Phase II Test Specification

Management(SNMP/MIBs)

Technical Document

Revision 1.0.3

IPv6 Forum *http://www.ipv6forum.org/*
IPv6 Logo Committee *http://www.ipv6ready.org/*
IPv6 Testing Lab, Chunghwa Telecom Labs (TW) *http://interop.ipv6.org.tw/*

MODIFICATION RECORD

Version	Date	Note
0.1.0	2006/03/31	SNMPv1 test specification
0.2.0	2006/08/07	SNMPv1 and MIB(4265,4266) merged
0.3.0	2006/08/07	SNMPv2C and MIB(4293) specifications
0.3.0.1	2006/10/04	1. Get subdivided into Get scalar and tabular 2. ifTable and udpTable selected for test tables
0.3.0.2	2007/04/12	1. Walk only the tables before that test 2. requestID was with fixed value for testing 3. the exact data type needs to be specified and exact message length calculated
0.3.0.3	2007/04/20	For Set operation cases, Get operations should be performed first
0.3.0.4	2007/05/15	1. Community name is public 2. NUT and TN error changed 3. tooBig response corrected(no silently discard) 4. Get, GetBulk, set replaced by GetRequest, GetBulkRequest, SetRequest
0.3.0.5	2007/06/29	1. Set placed after GetBulk(mandatory test items first rule) 2. All the terminologies shall follow RFC
0.3.0.6	2007/07/11	1. Roll back to the configuration before the successful set operation 2. Only sysUpTime and snmpTrapOID will be checked for trap testing
0.3.0.7	2007/08/21	Appendix added
0.4.0	2007/09/01	Testing Requirement Table added
0.4.1	2007/10/08	RFC revision history added
0.4.2	2007/11/07	1. advanced and basic to replace mandatory and optional items 2. set test cases revised 3. Title changed to “IPv6 Ready Logo Phase II Test Specification Management”
0.4.3	2007/11/30	1. Test Cases 31.v6SNMPv2C1.4.6.1Get with error-status type error 33.v6SNMPv2C1.4.6.3 Get with error-status non-zero error 34.v6SNMPv2C1.4.7.1 Get with error-index type error 36.v6SNMPv2C1.4.7.3 Get with error-index non-zero error, 77.v6SNMPv2C2.3.6.1 GetNext with error-status type error 79. v6SNMPv2C2.3.6.3 GetNext with error-status non-zero error 80.v6SNMPv2C2.3.7.1 GetNext with error-index type error 82.v6SNMPv2C2.3.7.3 GetNext with error-index non-zero error changed to Basic 2. Test v6SNMPv2C1.4.5.3.1 name changed to Get with requestID greater than maximum value(same change in v6SNMPv2C1.4.5.3.2, v6SNMPv2C2.3.5.3.1 v6SNMPv2C2.3.5.3.2) and these test cases are changed to Basic

		3. NA changed to bold font, Test Criteria changed to Test Requirement 4. Get-Response PDU all changed to Response-PDU, GetResponse changed to response 5. Table 4 RFC 4293 IP MIB – General Group Test Criteria IfAdminStatus and related description deleted 6. Set Operations renamed for better test naming to match with the SNMP test tool
0.4.4	2007/12/24	1. Added RFC 2580 and RFC 4001 in References 2. Moved v6SNMPv2C1.3 SNMPSilentDrops test case to v6SNMPv2CMIB1.2.2 SNMPSilentDrops 3. Added v6SNMPv2CMIB1.2.1 SNMPInPktsCountCheck test case. 4. Deleted NA in Table 4 5. Changed NA in Table 5 for ipv4InterfaceTable,ipv4InterfaceEntry and ipv4InterfaceIfInex and ipv6InterfaceIfIndex to B 6. Changed NA to B in IP statistics Table 6 Procedure 3 ipSystemStatsInOctets changed to 1.3.6.1.2.1.4.31.1.1.5.2 and ipSystemSttsOutOctets deleted Judgment OP2 : ipSystemStatsInOctets2 =ipSystmeStatsInOctets1+10*(40+8+40)+1*(40+8+43) 7. NA in Table 7 Required field is changed to B 8. NA in Table 8 Required field is changed to B 9. NA in Table 9 Required field is changed to B 10. NA in Table 10 Required field is changed to B 11. DefaultRouterTable is changed to Advanced 12. NA in Table 12 Required field is changed to B 13. NA in Table 13 Required field is changed to B 14. All icmpInMsgs Changed to icmpStasInMsgs
0.5.0	2008/3/6	Final inspection conducted and version frozen at 0.5.0 before going into public review
	2008/4/8	1. v6SNMPv2CMIB2.3.1(B) and v6SNMPv2CMIB1.10.1(A) are added and Table 1 is modified accordingly 2. NA in sysORTable is changed to A 3. v6SNMPv2CMIB2.8 Default Router Table is changed to A 4. icmp Statistics Table is changed to A 5. No detailed packet format in MIB test group 6. NICI deleted
0.5.1	2008/4/28	Extra RFC 1448 deleted in SNMPv2C revision history chart
0.5.2	2008/5/23	TrapOID length corrected
	2008/6/23	1. IP-MIB OIDs test criteria modified based on RFC 4293 Conformance and Compliance 2. mandatory groups included ipSystemStatsGroup, ipAddressGroup, ipNetToPhysicalGroup, ipDefaultRouterGroup, icmpStatsGroup and ipv6GeneralGroup,

		ipv6IfGroup,ipAddressPrefixGroup, ipv6ScopeGroup,ipv6RouterAdvertGroup 3. Host and Router column are added in all Tables in IP-MIB descriptions 4. Only IPv6 related are tested is added inv6SNMPv2CMIB2.1 General Objects Table 4(- means not to be tested) 5. ipAddressSpinLock in v6SNMPv2CMIB2.4 Internet Address Prefix Table is deleted and placed in v6SNMPv2CMIB2.5 Internet Address Table
0.5.3	2008/7/16	1. v6SNMPv2CMIB2.6 is changed to v6SNMPv2CMIB2.6.1(the original v6SNMPv2CMIB2.6) and v6SNMPv2CMIB2.6.2 where ipNetToPhysicalAddress are semantically checked after TN pings NUT as a REF-NODE
0.5.4	2008/8/21	1. v6SNMPv2CMIB2.6.2 IPNetToPhysicalAddress Check Procedure 5 TN performs another GetRequest with the OID value in Procedure 4 is changed Procedure TN performs another GetRequest with the OID value in Procedure 2 2. Remove Judgment 2
1.0.0	2008/12/16	1. Table 1 No.17 v6SNMPv2C 1.3.2.2.2 Get version number length error is changed to Get **with** version number length error 2. Table 1 No.18 v6SNMPv2C 1.3.2.2.3 Get version number value error is changed to Get **with** version number value error 3. Table 1 No.20 v6SNMPv2C 1.3.3.2 Get with community len error is changed to Get with community **length** error 4. Table 1 No.21 v6SNMPv2C 1.3.3.3.1 and No.64 v6SNMPv2C 2.3.3.3 Empty_community_string is changed to Empty community_string 5. Table 1 No.38 v6SNMPv2C 1.3.8.3.1 Get with FF value in variable binding name is changed to Get with FF value in variable-binding's name and No. 80 v6SNMPv2C 2.3.8.3.1 GetNext with FF value in variable binding name is changed to GetNext with FF value in variable-binding's name 6. Table 1 No.39 v6SNMPv2C 1.3.8.3.2 Get variable binding's value without NULL is changed to Get with variable-binding's value without NULL and No.81 v6SNMPv2C 2.3.8.3.2 GetNext variable binding's value without NULL is changed to GetNext with variable-binding's value without NULL 7. Table 1 No.40 v6SNMPv2C 1.3.8.3.3 Get with zero variable binding is changed to Get with zero variable-binding and No.82 v6SNMPv2C 2.3.8.3.3 GetNext with zero variable binding is changed to GetNext with zero variable-binding 8. Table 1 No.41 v6SNMPv2C 1.3.8.3.4 and No.83

		v6SNMPv2C 2.3.8.3.4 128_sub_identifier_check is changed to 128 sub_identifiers check 9. Table 1 No.66 community_string with carriageReturn_lineFeed is changed to community_string with CarriageReturn_LineFeed community_string 10. variable binding error in v6SNMPv2C1.3.8 and 2.3.8 are changed as variable-binding's error 11. Group 3 Tests for IPv6 SNMPv2C GetBulkRequest is changed to IPv6 SNMPv2C GetBulkRequest 12. Table 1 No. 103 v6SNMPv2C3.19 GetBulk with Large OID IID is changed to Index ID is changed to GetBulk with Large Index ID 13. Table 1 No. 109 v6SNMPv2C4.3.3 Set existent read-write objects with non-existent instance is changed to Set existent read-write object with non-existent instance 14. Table 1 No.110 v6SNMPv2C4.4 Set existent read-only object with existent instance is changed to bald font 15. Only ipAddressSpinLock is optional for test criteria is deleted in v6SNMPv2CMIB2.4 Internet Address Prefix Table 16. Fig 2 error is replaced with Fig 5 17. MAX-Access in v6SNMPv2CMIB2.5 Internet Address Table ipAddressIfIndex is changed to RW 18. variable bindings is changed to variable-bindings
1.0.2	2009/04/22	1. WARN explanation is added in v6SNMPv2C1.3.3.1 v6SNMPv2C1.3.5.1, v6SNMPv2C1.3.5.3.1, v6SNMPv2C1.3.5.3.2, v6SNMPv2C1.3.6.1, v6SNMPv2C1.3.6.3, v6SNMPv2C1.3.7.1, v6SNMPv2C1.3.7.3, v6SNMPv2C2.3.3.1, v6SNMPv2C2.3.5.1, v6SNMPv2C2.3.5.3.1, v6SNMPv2C2.3.5.3.2, v6SNMPv2C2.3.6.1, v6SNMPv2C2.3.6.3,v6SNMPv2C2.3.7.1 and v6SNMPv2C2.3.7.3
1.0.3	2010/05/13	1. v6SNMPv2C1.3.6.3 Get with error-status non-zero error, v6SNMPv2C1.3.7.3 Get with error-index non-zero error v6SNMPv2C2.3.6.3 GetNext with error-status non-zero and v6SNMPv2C2.3.7.3 GetNext with error-index non-zero error are changed to Advanced test case because the SNMP agent checks the name of a variable and not its value according to RFC3416 4.2 PDU Processing. To accommodate possible implementation of RFC1117, this test case will not be judged. 2. sysUpTime.0 in v6SNMPv2C2.3.6.3 GetNext with error-status non-zero error and v6SNMPv2C2.3.7.3 GetNext with error-index non-zero error is changed to sysUpTime so that the return OID will be the correct sysUpTime value
1.0.3	2010/06/02	1. sysUpTime in v6SNMPv2C2.3.6.3 GetNext with error-status

		non-zero error and v6SNMPv2C2.3.7.3 GetNext with error-index non-zero error is changed to sysUpTime.0 so that the returned OID is sysContact information

ACKNOWLEDGEMENTS

The IPv6 Forum would like to acknowledge the efforts of the following organizations in the development of this test suite.

Principle Authors

IPv6 Testing Lab, Chunghwa Telecom Labs (CHT-TL)

Commentators:
Yanick Pouffary (IPv6 Forum)
Erica Johnson (UNH-IOL)
Miyata Hiroshi (TAHI)
Daobiao Gong (BII)
Jan Safranek (redhat.com)

INTRODUCTION

Overview

The IPv6 forum plays a major role to promote the new generation of IP protocols by forming the IPv6 Ready Logo Committee that designs state-of-the-art interoperability platforms to help harmonize in the design, development and deployment of the new generation Internet Protocol version 6 (IPv6).

To provide the market a strong signal proving the level of interoperability across various products and to give confidence to users that IPv6 is currently operational, IPv6 Ready Logo Committee launched IPv6 Ready Logo Program in 2003 to contribute to the feeling that IPv6 is available and ready to be used.

After IPv6 Ready Logo Phase I, IPv6 Logo Program is now at its current Phase II activity providing more stringent IPv6 functionality test specifications for the IPv6 community.

To further provide verification for those IPv6 equipments' network management capabilities after their network layer functions are IPv6 Ready Logo certified, basic network management functions should also be tested as add-on features for managing these IPv6-capable intelligent nodes in the Internet.

The Simple Network Management Protocol (SNMP) is most commonly used protocol to communicate management information between the network management stations and the agents in the network elements.

According to the SNMP architectural model, network management stations execute management applications which monitor and control gateways, terminal servers and the like, and which have management agents for performing the network management functions requested by the network management stations.

How to verify the IPv6 network management functionality of the IPv6-capable equipment by SNMP protocol based on related RFCs is the main goal of this test suite. Various test demands are listed in this test suite to develop different test cases to verify and assure their network management functionalities.

Abbreviations and Acronyms

ASN.1	Abstract Syntax Notation One
MIB	Management Information Base
NUT	Node Under Test
PDU	Protocol Data Unit
SMI	Structure of Management Information
SNMPv2C	Simple Network Management Protocol Version 2 with Community Based
TCP	Transmission Control Protocol
TN	Testing Node
UDP	User Datagram Protocol

TEST ORGANIZATION

This document organizes tests by group based on related test methodology or goals. Each group begins with a brief set of comments pertaining to all tests within that group. This is followed by a series of description blocks; each block describes a single test. The format of the description block is as follows.

Purpose	This is the description of what the test case attempts to achieve.
Resource Requirements	External resources that can help facilitate the test.
Initialization	The Initialization part describes how to initialize and configure NUT and TN before starting the test. If no value is provided, then the default value shall be used.
Procedure	The Procedure will describe the step-by-step instructions for executing the test.
Judgment	The Judgment will describe the expected test results. NUT passes the test if description of the Judgment is observed.
References	This References section provides the specifications that are referred to in this documentation.

REQUIREMENTS

This test specification is intended to test the SNMPv2C protocol behaviors, as defined in RFC 3416, over IPv6. Interested parties shall acquire IPv6 Phase II Logo (for Core functionalities tests) first before they can conduct this advanced network management functions testing. For how to get an IPv6 Phase II Logo, please refer IPv6 Ready Test Specification Core Protocols.(http://www.ipv6ready.org/)

SNMPv2C, as defined in RFC 3416, is the most prevalent management protocol on top to IPv4 network layer. To provide a more efficient network management perspective, IETF's recent RFC 4293 document integrates RFC2465 for the IPv6 Management Information Base (MIB) and RFC 2466 for IPv6 ICMP MIB into Management Information Base for the Internet Protocol which will be used to manage network nodes in the IP Internet community using SNMP protocol.

Target SNMPv2C agent and manager

The following SNMPv2C tests shall be categorized as SNMPv2C agent or manager according to SNMPv2C agent or manager role the NUT undertakes. In this version, only SNMPv2C agent functions shall be tested.

Test Requirement

RFC 3416 (Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)), RFC 3418 (Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)) and RFC 2578, 2579 and 2580 are selected for this test specification version. Please see Fig. 1. for the SNMPv2C RFC revision history.

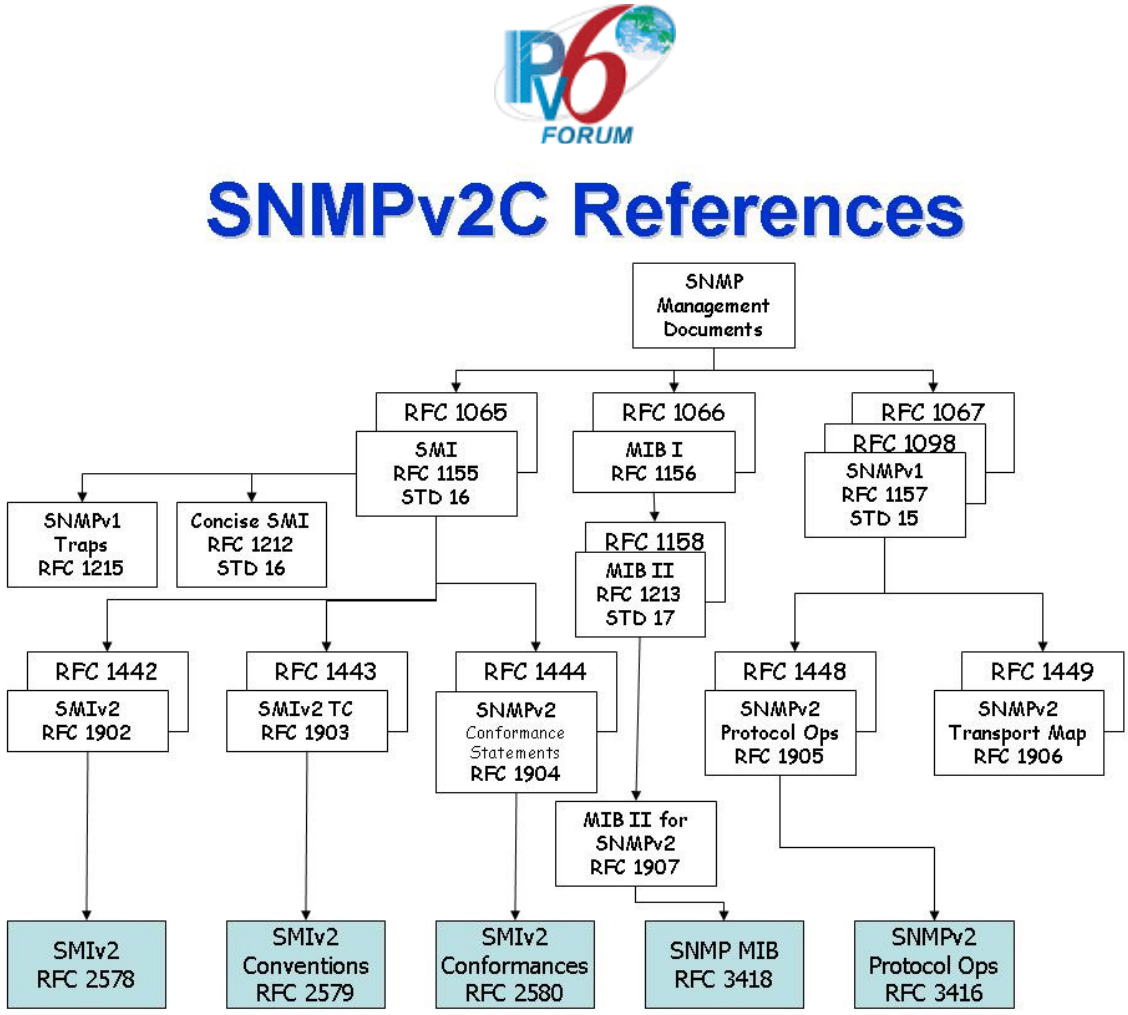


Fig. 1 The SNMPv2C Revision History and Relationship of MIB Standards

As to the SNMP MIB standard selection in this test specification, RFC 4293(IP MIB, Management Information Base for the Internet Protocol (IP)) is selected to reflect the most recent SNMP MIB standard after it obsoletes RFC 2011(SNMPv2 Management Information Base for the Internet Protocol using SMIv2) and Management Information Base for IP Version 6: ICMPv6 Group (RFC2466) and Management Information Base for IP Version 6: Textual Conventions and General Group (RFC 2465).

For the Interfaces Group MIB, RFC 2863(IF-MIB) is selected after it obsoletes RFC 2233 (The Interfaces Group MIB using SMIv2). As to Management Information Base for the Transmission Control Protocol (TCP), RFC 4022 TCP is selected after it obsoletes RFC 2452,RFC 2012.

Management Information Base for the User Datagram Protocol (UDP) (RFC 4113) is selected after it obsoletes RFC 2454,RFC 2013. For IP Tunnel MIB, RFC 4087(IP Tunnel MIB) obsoletes RFC 2997. Please see Fig. 2 for this SNMP MIB RFC revision history.

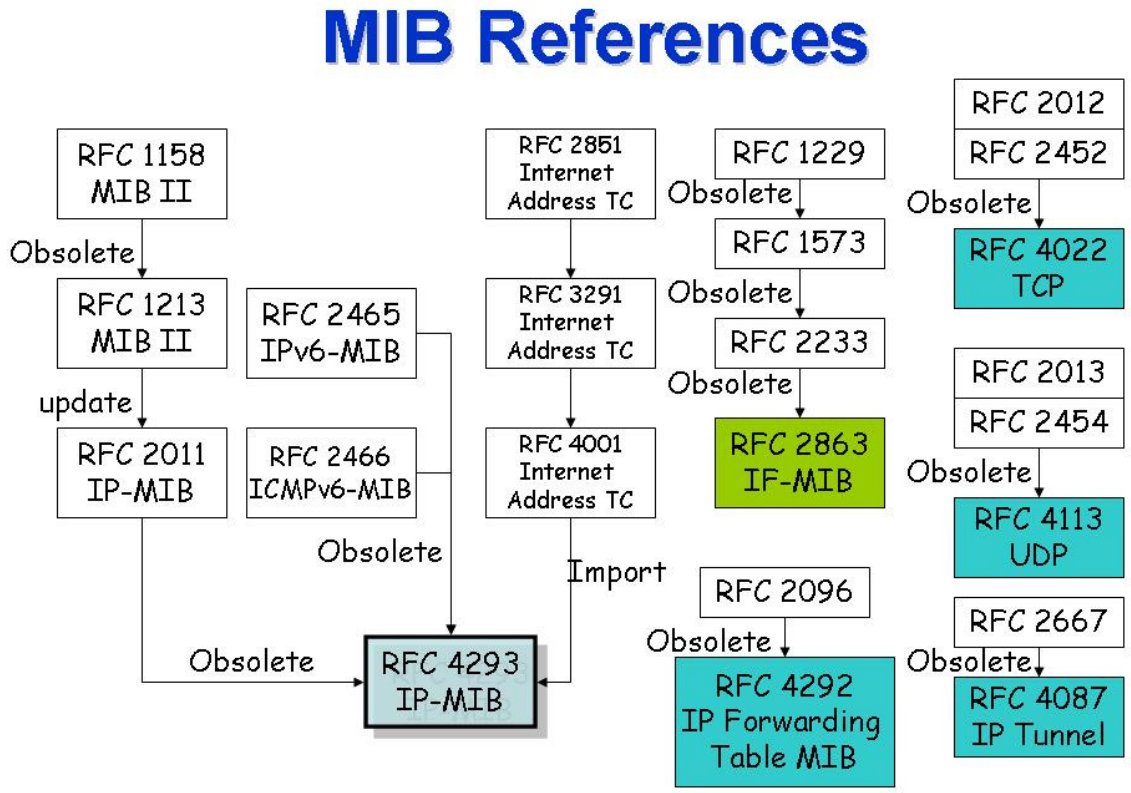


Fig. 2 The Revision History of SNMP MIB RFC Standards

Before starting this SNMP functionality testing, acquisition of IPv6 Ready Phase II Logo Core is the prerequisite. In this specification, SNMPv2C (RFC 3416) is used as the SNMP protocol. RFC 3418 and 4293 IP-MIB are selected for this test MIB requirement. Future inclusions of more MIBs including SNMPv3 will be planned in the next phase test plans. Fig. 3 is the relationship of IPv6 Ready Logo Phase II Core and SNMP/MIBs.

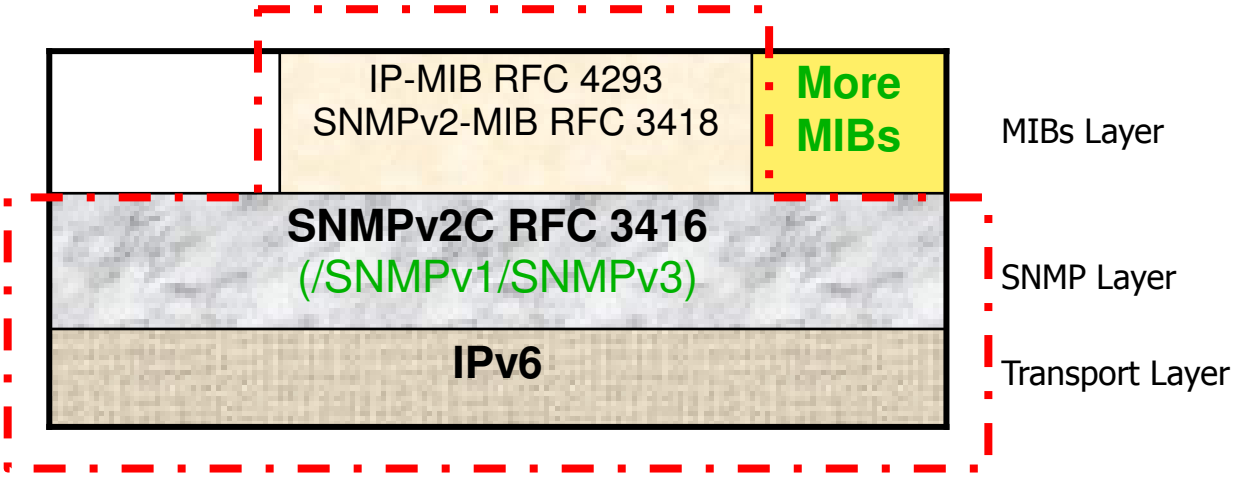


Fig. 3 The Relationship of IPv6 Ready Logo Phase II Core and SNMP/MIBs

SNMPv3 whose revision history is depicted in Fig. 4 will not be covered in this specification.

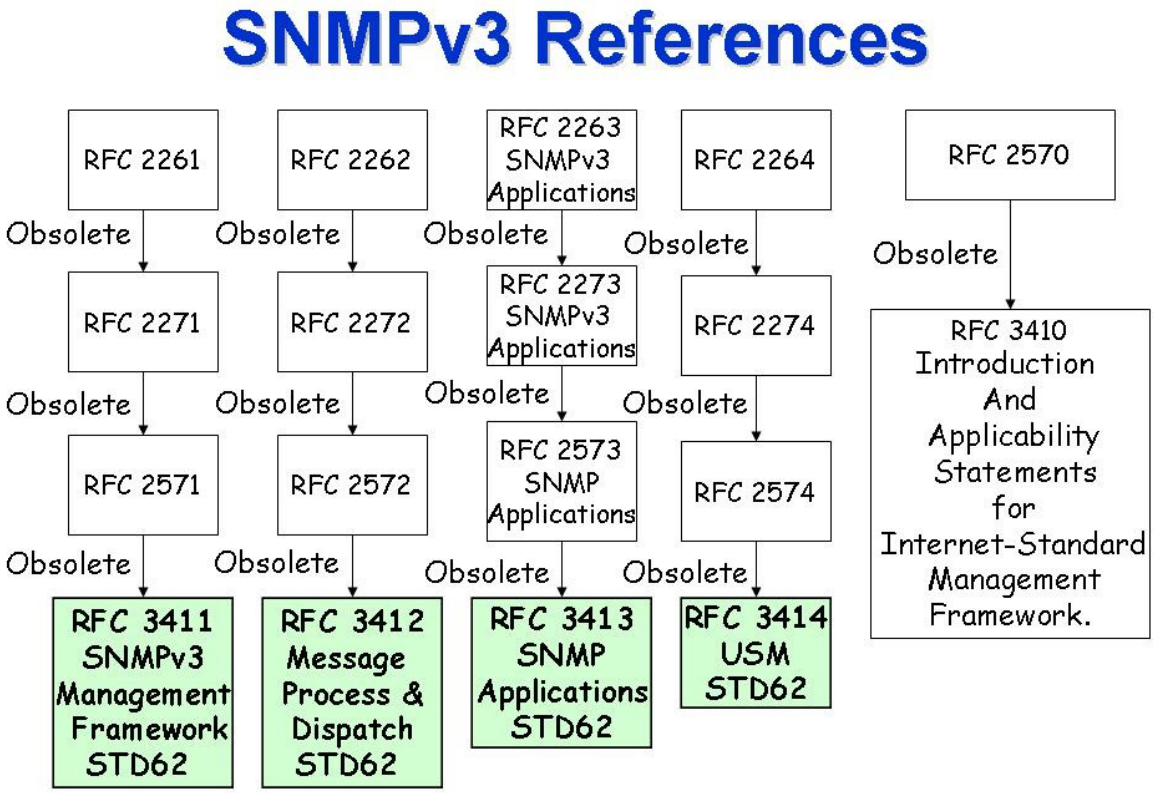


Fig. 4 Revision History of SNMPv3 RFC Standards

Table 1 is the list of IPv6 Ready Logo SNMP Conformance Test Requirement. Only the basic test items are mandatory and shall be judged for the test results. Advanced items are optional.

Table 1 IPv6 Ready Logo SNMP/MIBs Conformance Test Requirement

Note: B stands for Basic, A for Advanced

No	Test Item	Test Name	Required
1.	Pre-test		B
RFC 3416 SNMPv2C Protocol Operations			
Group 1 IPv6 SNMPv2C GetRequest			
	v6SNMPv2C1.1	**Get Operations**	
	v6SNMPv2C1.1.1	**Get scalar object**	
2.	v6SNMPv2C1.1.1.1	Get single scalar object with correct values	B
3.	v6SNMPv2C1.1.1.2	Get single scalar object with noSuchObject	B
4.	v6SNMPv2C1.1.1.3	Get single scalar object with noSuchInstance	B
5.	v6SNMPv2C1.1.1.4	Get multiple scalar objects with correct values	B
6.	v6SNMPv2C1.1.1.5	Get multiple scalar objects with noSuchObject, noSuchInstance	B
	v6SNMPv2C1.1.2	**Get tabular objects**	
	v6SNMPv2C1.1.2.1	**Get OIDs from the same table**	
7.	v6SNMPv2C1.1.2.1.1	Get OIDs from the same table with correct values	B
8.	v6SNMPv2C1.1.2.1.2	Get OIDs from the same table with noSuchObject	B
9.	v6SNMPv2C1.1.2.1.3	Get OIDs from the same table with noSuchInstance	B
	v6SNMPv2C1.1.2.2	**Get OIDs from different tables**	
10.	v6SNMPv2C1.1.2.2.1	Get OIDs from different tables with correct values	B
11.	v6SNMPv2C1.1.2.2.2	Get OIDs from different tables with noSuchObject	B
12.	v6SNMPv2C1.1.2.2.3	Get OIDs from different tables with noSuchInstance	B
13.	**v6SNMPv2C1.2**	**Get RequestID Correlation Check**	B
	v6SNMPv2C1.3	**Error Check**	
	v6SNMPv2C1.3.1	**Get with sequence_of error**	
14.	v6SNMPv2C1.3.1.1	Get with sequence_of type error	B
15.	v6SNMPv2C1.3.1.2	Get with sequence_of length error	B
	v6SNMPv2C1.3.2	**Get with version number error**	
16.	v6SNMPv2C1.3.2.1	Get with version number type error	B
17.	v6SNMPv2C1.3.2.2	Get with version number length error	B
18.	v6SNMPv2C1.3.2.3	Get with version number value error	B
	v6SNMPv2C1.3.3	**Get with community error**	
19.	v6SNMPv2C1.3.3.1	Get with community type error	B
20.	v6SNMPv2C1.3.3.2	Get with community length error	B
	v6SNMPv2C1.3.3.3	**Get with community value error**	
21.	v6SNMPv2C1.3.3.3.1	Empty community_string	B

22.	v6SNMPv2C1.3.3.3.2	Inconsistent community_string	B
23.	v6SNMPv2C1.3.3.3.3	community_string with CarriageReturn LineFeed	B
	v6SNMPv2C1.3.4	**Get with PDU error**	
24.	v6SNMPv2C1.3.4.1	Get with PDU type error	B
25.	v6SNMPv2C1.3.4.2	Get with PDU length error	B
	v6SNMPv2C1.3.5	**Get with request ID error**	
26.	v6SNMPv2C1.3.5.1	Get with request ID type error	B
27.	v6SNMPv2C1.3.5.2	Get with request ID length error	B
	v6SNMPv2C1.3.5.3	**Get with request ID value error**	
28.	v6SNMPv2C1.3.5.3.1	Get with requestID greater than maximum value(214783647,0x0CCD569F)	A
29.	v6SNMPv2C1.3.5.3.2	Get with requestID smaller than minimum value(-14783648,0xF332A960)	A
	v6SNMPv2C1.3.6	**Get with error-status error**	
30.	v6SNMPv2C1.3.6.1	Get with error-status type error	B
31.	v6SNMPv2C1.3.6.2	Get with error-status length error	B
32.	v6SNMPv2C1.3.6.3	Get with error-status non-zero error	A
	v6SNMPv2C1.3.7	**Get with error-index error**	
33.	v6SNMPv2C1.3.7.1	Get with error-index type error	B
34.	v6SNMPv2C1.3.7.2	Get with error-index length error	B
35.	v6SNMPv2C1.3.7.3	Get with error-index non-zero error	A
	v6SNMPv2C1.3.8	**Get with variable-binding error**	
36.	v6SNMPv2C1.3.8.1	Get with OID type error	B
37.	v6SNMPv2C1.3.8.2	Get with OID length error	B
	v6SNMPv2C1.3.8.3	**Get with OID value error**	
38.	v6SNMPv2C1.3.8.3.1	Get with FF value in variable-binding's name	B
39.	v6SNMPv2C1.3.8.3.2	Get variable-binding's value without NULL	B
40.	v6SNMPv2C1.3.8.3.3	Get with zero variable-bindings	B
41.	v6SNMPv2C1.3.8.3.4	128 sub-identifiers check	B
42.	v6SNMPv2C1.3.9	Get with tooBig message	B
Group 2 IPv6 SNMPv2C GetNextRequest			
	v6SNMPv2C2.1	**GetNext Operations**	
	v6SNMPv2C2.1.1	**GetNext scalar object**	
43.	v6SNMPv2C2.1.1.1	GetNext single scalar object	B
44.	v6SNMPv2C2.1.1.2	GetNext single scalar object from non-existent object	B
45.	v6SNMPv2C2.1.1.3	GetNext single scalar object from existent instance	B
46.	v6SNMPv2C2.1.1.4	GetNext single scalar object from non-existent instance	B
47.	v6SNMPv2C2.1.1.5	GetNext from 2.0 (endOfMIBView)	B
48.	v6SNMPv2C2.1.1.6	GetNext multiple scalar objects	B
	v6SNMPv2C2.1.2	**GetNext tabular objects**	
49.	v6SNMPv2C2.1.2.1	GetNext from ifTable	B
50.	v6SNMPv2C2.1.2.2	GetNext from ifEntry	B
51.	v6SNMPv2C2.1.2.3	GetNext from ifIndex	B

52.	v6SNMPv2C2.1.2.4	GetNext from ifIndex.0	B
53.	v6SNMPv2C2.1.2.5	GetNext from ifIndex.10000	B
54.	v6SNMPv2C2.1.2.6	GetNext tabular objects with multiple OIDs	B
55.	v6SNMPv2C2.1.2.7	GetNext multiple OIDs from different tables	B
56.	**v6SNMPv2C2.2**	**GetNext RequestID Correlation Check**	B
	v6SNMPv2C2.3	**Error Check**	
	v6SNMPv2C2.3.1	**GetNext with sequence_of error**	
57.	v6SNMPv2C2.3.1.1	GetNext sequence_of type error	B
58.	v6SNMPv2C2.3.1.2	GetNext with sequence_of length Error	B
	v6SNMPv2C2.3.2	**GetNext with version number error**	
59.	v6SNMPv2C2.3.2.1	GetNext with version number type error	B
60.	v6SNMPv2C2.3.2.2	GetNext with version number length error	B
61.	v6SNMPv2C2.3.2.3	GetNext with version number value error	B
	v6SNMPv2C2.3.3	**GetNext with community error**	
62.	v6SNMPv2C2.3.3.1	GetNext with community type error	B
63.	v6SNMPv2C2.3.3.2	GetNext with community length error	B
	v6SNMPv2C2.3.3.3	**GetNext with community value error**	
64.	v6SNMPv2C2.3.3.3.1	Empty community_string	B
65.	v6SNMPv2C2.3.3.3.2	Inconsistent community_string	B
66.	v6SNMPv2C2.3.3.3.3	community_string_with_CarriageReturnLineFeed	B
	v6SNMPv2C2.3.4	**GetNext with PDU error**	
67.	v6SNMPv2C2.3.4.1	GetNext with PDU length error	B
	v6SNMPv2C2.3.5	**GetNext with request ID error**	
68.	v6SNMPv2C2.3.5.1	GetNext with request ID type error	B
69.	v6SNMPv2C2.3.5.2	GetNext with request ID length error	B
	v6SNMPv2C2.3.5.3	**GetNext with request ID value error**	
70.	v6SNMPv2C2.3.5.3.1	GetNext with requestID greater than maximum value(214783647,0x0CCD569F)	A
71.	v6SNMPv2C2.3.5.3.2	GetNext with requestID smaller than minimum value(-214783648, F332A960)	A
	v6SNMPv2C2.3.6	**GetNext with error-status error**	
72.	v6SNMPv2C2.3.6.1	GetNext with error-status type error	B
73.	v6SNMPv2C2.3.6.2	GetNext with error-status length error	B
74.	v6SNMPv2C2.3.6.3	GetNext with error-status non-zero error	A
	v6SNMPv2C2.3.7	**GetNext with error-index error**	
75.	v6SNMPv2C2.3.7.1	GetNext with error-index type error	B
76.	v6SNMPv2C2.3.7.2	GetNext with error-index length error	B
77.	v6SNMPv2C2.3.7.3	GetNext with error-index non-zero error	A
	v6SNMPv2C2.3.8	**GetNext with variable-bindings error**	
78.	v6SNMPv2C2.3.8.1	GetNext with OID type error	B
79.	v6SNMPv2C2.3.8.2	GetNext with OID length error	B
	v6SNMPv2C2.3.8.3	**GetNext with OID value error**	
80.	v6SNMPv2C2.3.8.3.1	GetNext with FF value in variable-binding's name	B
81.	v6SNMPv2C2.3.8.3.2	GetNext with variable-binding's value without NULL	B

82.	v6SNMPv2C2.3.8.3.3	GetNext with zero variable-bindings	B
83.	v6SNMPv2C2.3.8.3.4	128 sub-identifier check	B
84.	**v6SNMPv2C2.4**	**GetNext with tooBig message**	B
Group 3 IPv6 SNMPv2C GetBulkRequest			
85.	v6SNMPv2C3.1	GetBulk with zero non-repeaters, zero max-repetitions and zero variable-bindings	B
86.	v6SNMPv2C3.2	GetBulk with zero non-repeaters, non-zero max-repetitions and zero variable-bindings	B
87.	v6SNMPv2C3.3	GetBulk with non-zero non-repeaters, zero max-repetitions and zero variable-bindings	B
88.	v6SNMPv2C3.4	GetBulk with non-zero non-repeaters, non-zero max-repetitions and zero variable-bindings	B
89.	v6SNMPv2C3.5	GetBulk with zero non-repeaters, zero max-repetitions and non-zero variable-bindings	B
90.	v6SNMPv2C3.6	GetBulk with non-zero non-repeaters, zero max-repetitions and non-zero variable-bindings	B
91.	v6SNMPv2C3.7	GetBulk with zero non-repeaters, non-zero max-repetitions and non-zero variable-bindings	B
92.	v6SNMPv2C3.8	GetBulk with non-zero non-repeaters, non-zero max-repetitions and non-zero variable-bindings	B
93.	v6SNMPv2C3.9	GetBulk with negative non-repeaters, zero max-repetitions and zero variable-bindings	B
94.	v6SNMPv2C3.10	GetBulk with zero non-repeaters, negative max-repetitions and zero variable-bindings	B
95.	v6SNMPv2C3.11	GetBulk with negative non-repeaters, negative max-repetitions and zero variable-bindings	B
96.	v6SNMPv2C3.12	GetBulk with zero non-repeaters, negative max-repetitions and non-zero variable-bindings	B
97.	v6SNMPv2C3.13	GetBulk with negative non-repeaters, zero max-repetitions and non-zero variable-bindings	B
98.	v6SNMPv2C3.14	GetBulk with negative non-repeaters, negative max-repetitions and non-zero variable-bindings	B
99.	v6SNMPv2C3.15	GetBulk with large max-repetitions	B
100.	v6SNMPv2C3.16	GetBulk with non-repeaters greater than variable-bindings	B
101.	v6SNMPv2C3.17	GetBulk with non-repeaters less than variable-bindings	B
102.	v6SNMPv2C3.18	GetBulk with 128 sub-identifiers	B
103.	v6SNMPv2C3.19	GetBulk with Large Index ID	B
104.	v6SNMPv2C3.20	GetBulk with Different Tables	B
Group 4 IPv6 SNMPv2C SetRequest			
105.	v6SNMPv2C4.1	Set non-existent object	A
106.	v6SNMPv2C4.2	Set existent read-write objects	A
	v6SNMPv2C4.3	**Set existent read-write objects error**	
107.	v6SNMPv2C4.3.1	Set with wrongType	A
108.	v6SNMPv2C4.3.2	Set with wrongValue	A
109.	v6SNMPv2C4.3.3	Set existent read-write object with non-existent	A

		instance	
110.	v6SNMPv2C4.4	Set existent read-only object with existent instance	A
	v6SNMPv2C4.5	**Set multiple variables**	
111.	v6SNMPv2C4.5.1	Set two read-write variables	A
112.	v6SNMPv2C4.5.2	Set two read-write variables with wrong type of the second variable	A
113.	v6SNMPv2C4.5.3	Set two read-write variables with wrong type of the first variable	A
114.	v6SNMPv2C4.5.4	Set two read-write variables with wrong type of the variables	A
115.	v6SNMPv2C4.5.5	Set read-write and read-only variables	A
116.	v6SNMPv2C4.5.6	Set read-write variable with wrong type and read-only variable	A
117.	v6SNMPv2C4.5.7	Set read-only and read-write variables	A
Group 5 IPv6 SNMPv2C Trap			
118.	v6SNMPv2C5.1	Trap Test	B
RFC 3418 SNMPv2 MIB			
119.	v6SNMPv2CMIB1.1	System Group	B(only B item in Table 2)
120.	v6SNMPv2CMIB1.2	SNMP Group	A
121.	v6SNMPv2CMIB1.2.1	snmpInPkts counter check	A
122.	v6SNMPv2CMIB1.2.2	snmpSilentDrops counter check	A
RFC 4293 IP MIB			
123.	v6SNMPv2CMIB2.1	General Objects	B(only B item in Table 2)
124.	v6SNMPv2CMIB2.2	InterfaceTables	B(only B item in Table 2)
125.	v6SNMPv2CMIB2.3	IP Statistics Tables	B(A for ipIfStatsTable)
126.	v6SNMPv2CMIB2.3.1	ipSystemStatsInOctets counter check	B
127.	v6SNMPv2CMIB2.4	Internet Address Prefix Table	B
128.	v6SNMPv2CMIB2.5	Internet Address Table	B
129.	v6SNMPv2CMIB2.6.1	Internet Address Translation Table	B
130.	v6SNMPv2CMIB2.6.2	ipNetToPhysicalAddress Check	B
131.	v6SNMPv2CMIB2.7	IPv6 Scope Zone Index Table	B
132.	v6SNMPv2CMIB2.8	Default Router Table	B
133.	v6SNMPv2CMIB2.9	IPv6 Router Advertisement Table(for IPv6 Routers only)	B
134.	v6SNMPv2CMIB2.10	ICMP Statistics Table	B
135.	v6SNMPv2CMIB2.10.1	icmpStatInMsgs counter check	B

REFERENCES

The following documents are referenced in this text

[ADDR] R. Hinden, S. Deering, Internet Protocol Version 6 (IPv6) Addressing Architecture, RFC 3315, April 2003.

[ICMPv6] A. Conta, S. Deering, M. Gupta, Ed., Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification, RFC 4443, March 2006.

[INA-TC] M. Daniele, B. Haberman, S. Routhier, J. Schoenwaelder, Textual Conventions for Internet Network Addresses, RFC 4001, February 2005.

[IPv6-SPEC] Hinden, R., S. Deering, Internet Protocol, version 6 (IPv6) Specification, RFC 2460, December 1998.

[IPv6 Ready Phase II Test Specification Core Protocols] IPv6 Forum IPv6 Ready Technical Documentation. (http://www.ipv6ready.org/)

[MIB-DEF] M. Rose, K. McCloghrie, Concise MIB definitions, RFC 1212, March 1991.

[MIB-II] K. McCloghrie, M. Rose, Management Information Base for Network Management of TCP/IP-based internets MIB-II, RFC 1213, March 1991.

[MIB for IP] Shawn A. Routhier, Management Information Base for the Internet Protocol (IP), RFC 4293, April 2006.

[SIMI] M. Rose, K. McCloghrie, Structure and Identification of Management Information for TCP/IP-based Internets, RFC 1155, May 1990.

[SMIv2] K. McCloghrie, D. Perkins, J. Schoenwaelder, Structure of Management Information version 2 (SMIv2), RFC 2578, April 1999.

[SMIv2-CS] K. McCloghrie, D. Perkins, J. Schoenwaelder, Conformance Statements for SMIv2. RFC 2580, April 1999.

[SMIv2-TC] K. McCloghrie, D. Perkins, J. Schoenwaelder, Textual Conventions for SMIv2, RFC 2579, April 1999.

[SNMPv1] J. Case, M. Fedor, M. Schoffstall, J. Davin, A Simple Network Management Protocol (SNMPv1), RFC 1157, May 1990.

[SNMP] William Stallings, SNMP, SNMPv2, SNMPv3, and RMON 1 and 2, 3rd ed., 1999.

[SNMPv2] J. Case, K.McCloghire, M.Rose S. Waldbusser , Protocol Operations for version 2 of the Simple Network Management Protocol, RFC 3416, December 2002.

[SNMPv2C] J. Case, K.McCloghire, M.Rose S. Waldbusser , Introduction to Community-based SNMPv2, RFC 1901, January 1996.

[SNMPv2-MIB] Management Information Base (MIB) for the Simple Network Management Protocol (SNMP), RFC 3418, December 2002.

[SMIv2] K. McCloghrie, D. Perkins, J. Schoenwaelder, Structure of Management Information version 2 (SMIv2), RFC 2578, April 1999.

TABLE OF CONTENTS

Figures and Tables

Common Topology

Testing Architecture

The network topology as shown in Fig. 5 shall be used for all tests in this test suite. TN (Test Node) and NUT (Node under Test) can use either link-local address (when in the same LAN environment) or global address (when in the same LAN segment or in the Internet environment) to talk to each other.

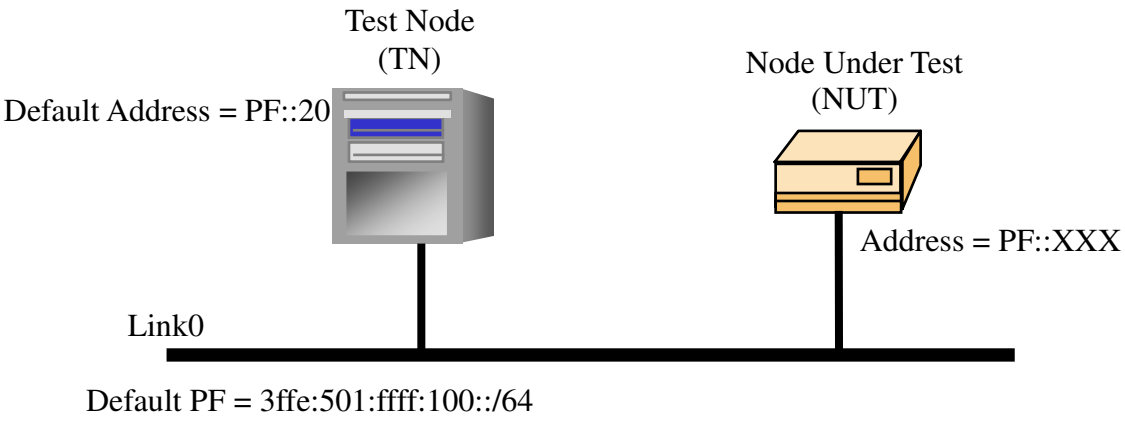


Fig. 5 Test Architecture

Common Test Setup

Tests in this test specification may refer to the Common Test Setup Procedure as shown in Fig. 6 defined in this section.

Common Test Setup Procedure
This minimal setup procedure shall refer the testing architecture described in this test suite. NUT must decide to assume either SNMPv2C manager or SNMPv2C agent role before conduction this SNMPv2C test. The communication link between TN and NUT must be ok before starting this SNMPv2C test. The timer value for waiting for a SNMPv2C response is set to be 30 seconds.

For NUT which is functioning as router, please first configure the router address and default route and then TN performs ping NUT with link-local and global addresses to make sure the IPv6 communication link is ready before starting the test.

For NUT which is functioning as host, TN, emulating REF-NODE, will send Router Advertisement to NUT and NUT will then obtain IPv6 prefix information for the test. Ping operations should be followed to make sure the IPv6 link is ready.

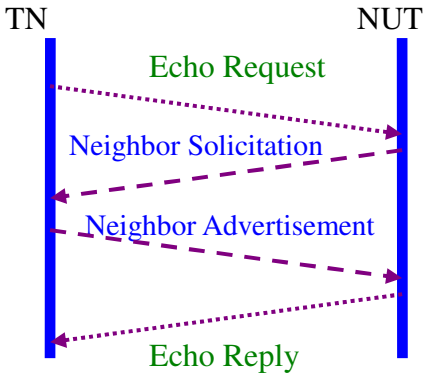


Fig. 6 Common IPv6 Link Test Setup Basic Before SNMPv2C Testing

The following procedure describes the common test setup before the SNMPv2C testing

a. TN sends “Echo Request” to NUT
b. TN waits for “Echo Reply” from NUT
c. If ”Neighbor Solicitation” message is received first, NUT will respond ”Neighbor Advertisement” and wait for ”Echo Reply”
d. TN receives ”Echo Reply”

“Echo Request” message format is as follows：
IPv6 Header
Version = 6
Traffic Class = 0
FlowLabel = 0
PayloadLength = 16
NextHeader = 58 (ICMP)

SourceAddress = Tester Address
DestinationAddress = Target Address

ICMP Echo Request
Type = 128 (Echo Request)
Code = 0
Checksum = (auto)
Identifier = 0xffffff
SequenceNumber = 1
PayloadData = {1,2,3,4,5,6,7,8}

Expected ”Echo Reply” format is as follows

IPv6 Header
Version = 6
Traffic Class = 0
Flow Label = 0
Payload Length = 16
Next Header = 58 (ICMP)
Source Address = Target Address
Destination Address = Tester Address

ICMP Echo Reply
Type = 129 (Echo Reply)
Code = 0
Checksum = (auto)
Identifier = 0xffffff (same as Echo Request)
Sequence Number = 1 (same as Echo Request)
Payload Data = {1,2,3,4,5,6,7,8} (same as Echo Request)

Common Default Configuration values and Pre-Test/Post-Test Procedures(for all tests)

The version of SNMPv2C packets between TN and NUT is version 2 and the community object identifier of SNMPv2C message is public. The UDP port of SNMPv2C agent is 161 and the trap SNMPv2C UDP port is 162. TN communicates with NUT via SNMPv2C over UDP over IPv6.

Default Packets

The SNMPv2C packets are carried over IPv6 link with the following formats.

IPv6 Header

version (6)
Traffic Class (0)
Flow Label (0)
Payload length (*)
Next Header UDP (0x11)
Hop Limit (0x40)
Source Address (TN’s Address)
Destination Address (NUT’s Address)

UDP

Source Port/Destination Port (161)
Length
checksum

SNMPv2C PDU

version =1 (version 2)
Community = Public
SNMPv2C PDU

The SNMPv2C PDU general formats follow the RFC 3416 descriptions. Please see the following RFC 3416 section for more detailed descriptions.

SNMPv2C PDU

PDU Type (Get/GetNext/Set/Response)
Request ID
Error Status
Error Index
variable-bindings

SNMPv2C GetBulkRequest PDU

PDU Type (GetBulkRequest)
Request ID
non-repeaters
max-repetitions
variable-bindings (object1/value1 object2/value2…..)

Section 1 Tests For NUT as SNMPv2C Agent
RFC 3416 SNMPv2(SNMPv2C) Protocol Operations

Scope

Any IPv6 capable SNMPv2C agent can be targets for this SNMPv2C test specification

Overview

This test specification is designed to test SNMPv2C functionalities based in RFC 3416, IETF specification by which management information for a network element may be inspected or altered by logically remote users. The following recaps the basic SNMP operations in RFC3416.

Default Packets

In SNMPv2C, information is still exchanged between a management station and an agent in the form of a SNMPv2C message. Each message includes a version number indicating the version of SNMPv2C, a community object identifier to be used for this exchange, and one of 8 different types of protocol data units which include GetRequest, GetNextRequest, GetBulkRequest, SetRequest, Response, InformRequest, Report and SNMPv2-Trap PDU. GetBulkRequest, InformRequest, Report and SNMPv2-Trap PDUs are new compared to those of SNMPv1 as defined in RFC 1157. The following paragraphs recap those SNMPv2 PDUs. For detailed description of SNMPv1 PDUs, please see RFC 1157.

A GetBulkRequest-PDU is generated and transmitted at the request of an application. The purpose of the GetBulkRequest-PDU is to request the transfer of a potentially large amount of data, including, but not limited to, the efficient and rapid retrieval of large tables.

An InformRequest-PDU is generated and transmitted by a SNMPv2 entity on behalf of a notification originator application. The InformRequest-PDU is often used to notify a notification receiver application that an event has occurred or that a condition is present. This is a confirmed notification delivery mechanism, although there is, of course, no guarantee of delivery. The InformRequest will not be tested in this test specification.

A SNMPv2-Trap-PDU is generated and transmitted by an SNMP entity on behalf of a notification originator application. The SNMPv2-Trap-PDU is often used to notify a notification receiver application at a logically remote SNMPv2C entity that an event has occurred or that a condition is present. There is no confirmation associated with this notification delivery mechanism. Only Trap conditions with cold start and Link Up/Link Down are selected to be tested for the TrapRequest PDU test function.

SNMPv2 has extended the protocol operations error status to include noAccess, wrongType, wrongLength, wrongEncoding, wrongvalue, noCreation, inconsistentalue, resourceUnavailable, commitFailed, undoFailed authorizationError, notWritable, inconsistentName in addition to the SNMPv1 noError, tooBig, noSuchName, badValue error status codes.

Pre-Test

Purpose

Verify that NUT playing the SNMPv2C agent is functioning normally before the test. TN will send SNMPv2C Get sysUpTime to NUT. No tests shall be followed on failure of this pre-test.

Resource Requirements

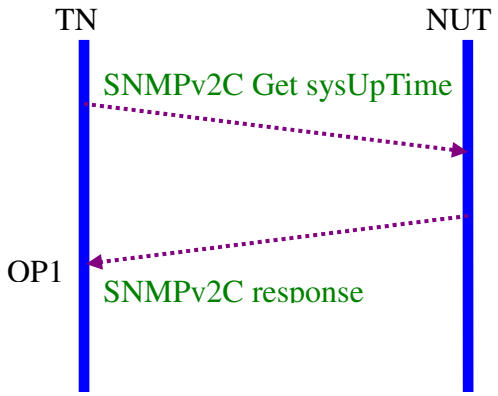
- Packet generator
- Monitor to capture packets

Initialization

Network Topology: Please refer Fig 5. Test Architecture.
Setup: Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C Get scalar object to NUT by issuing SNMPv2C Get to get sysUpTime 1.3.6.1.2.1.1.3 in system group in MIB II.
2. NUT replies SNMPv2C Response with correct SysUpTime value to TN

1st Packet(sent by TN)

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address		TN_ADDRESS			
	Destination Address		NUT_ADDRESS			
UDP Header	Source Port		any			
	Destination Port		161			
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)			
			type	len	value	
			30	26		
	Version	1(SNMPv2C)	02	01	01	
	community	public	04	06	70 75 62 6C 69 63	
	Data PDU type	GetRequest	A1	19		
	Data request-id	12	02	01	0C	

	t a	error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	Version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
	value	*	43	*	TimeTicks

Note * indicates variable values that vary according with the actual packet and variable binding

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1:
TN received SNMPv2C response from NUT responding to SNMPv2C Get object request correctly

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be zero
4. value field is the sysUpTime in system group of NUT with correct syntax type and value within the defined range field

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

Group 1 Tests for IPv6 SNMPv2C GetRequest

Scope

The following tests verify the GetRequest commands in the SNMPv2C protocol.

Overview

Tests in this group verify that a SNMPv2C agent node can properly process and generate the correct SNMPv2C messages with Response PDUs according to the SNMPv2C Get commands from the SNMPv2C manager. These tests also verify a SNMPv2C agent node will transmit the appropriate SNMPv2C parameter problem error messages in response to invalid or unknown fields in the received SNMPv2C packets.

v6SNMPv2C1.1 Get Operations
v6SNMPv2C1.1.1 Get scalar object
v6SNMPv2C1.1.1.1 Get single scalar object with correct values

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest object packet from the SNMPv2C manager.

Resource Requirements

•Packet generator
•Monitor to capture packets

Initialization

Network Topology

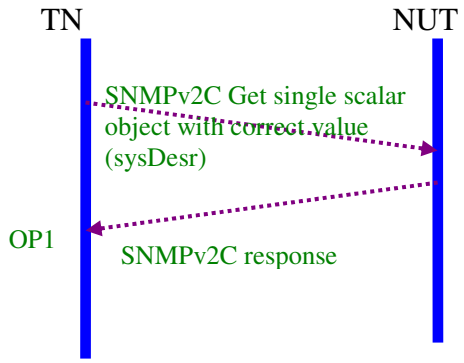
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest scalar object to NUT by issuing SNMPv2C GetRequest to get sysDescr 1.3.6.1.2.1.1.1 in system group in MIB II
2. NUT replies SNMPv2C Response with correct values to TN

1st Packet(sent by TN)

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	26	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63

	D a t a	PDU type		GetRequest	A0	19	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data: PDU type	Response	A2	*	
	Data: request-id	12	02	01	12
	Data: error-status	0	02	01	00
	Data: error-index	0	02	01	00
	Data:		30	*	
	Data: variable-bindings		30	*	
	Data: variable-bindings: name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
	Data: variable-bindings: value	octet string of NUT system description	04	*	variable string*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1:
TN received SNMPv2C response from NUT responding to SNMPv2C Get scalar

object request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal to zero
4. value field is the system descriptor in system group of NUT with correct syntax type and value within the defined range field

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.1.2 Get single scalar object with noSuchObject

Purpose

Verify that SNMPv2C agent can properly detect the GetRequest with non-existent OID error packet from SNMPv2C manager.

Resource Requirements

•Packet generator
•Monitor to capture packets

Initialization

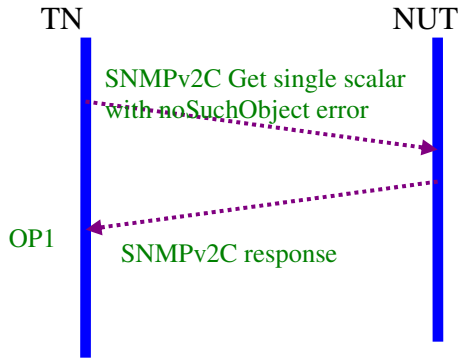
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with non-existent error OID to test NUT.
2. NUT returns Response with noSuchObject.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	26	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	GetRequest	A0	19	
	Data request-id	12	02	01	0C
	Data error-status	0	02	01	00
	Data error-index	0	02	01	00
	Data		30	0E	

		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.255	06	08	2b 06 01 02 01 01 81 7F
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			NUT_ADDRESS		
	Destination Address			TN_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.255	06	08	2b 06 01 02 01 01 81 7F
			value	noSuchObject	80	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will return Response with noSuchObject.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C

GetRequest

3. error-status must be equal to zero and error-index must be equal zero
4. value field is the noSuchObject

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.1.3 Get single scalar object with noSuchInstance

Purpose

Verify that SNMPv2C agent can properly detect the GetRequest with error OID packet from SNMPv2C manager.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

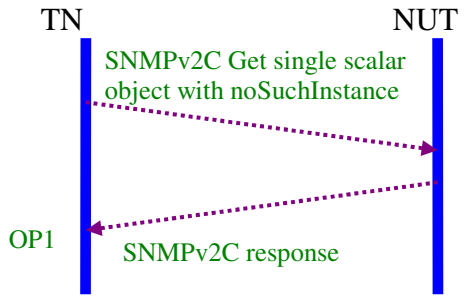
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with error OID to test noSuchInstance response from NUT.
2. NUT returns Response with NoSuchInstance.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	26	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data: PDU type	GetRequest	A0	19	
	Data: request-id	12	02	01	0C
	Data: error-status	0	02	01	00
	Data: error-index	0	02	01	00
	Data:		30	0E	

		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.1.1	06	08	2b 06 01 02 01 01 01 01
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			NUT_ADDRESS		
	Destination Address			TN_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.1.1	06	08	2b 06 01 02 01 01 01 01
			value	noSuchInstance	81	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will return Response with noSuchInstance.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest

3. error-status must be equal to zero and error-index must be equal zero
4. value field is the noSuchInstance

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.1.1.4 Get multiple scalar objects with correct values

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest multiple objects packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

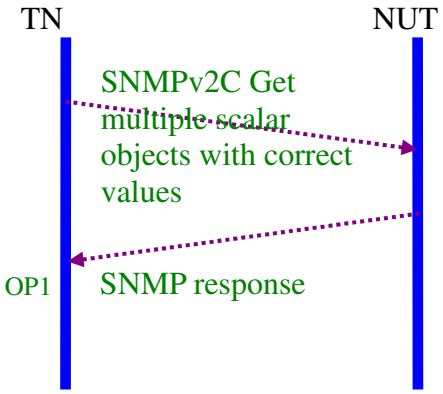
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest scalar object to NUT by issuing SNMPv2C GetRequest to get all scalar OID in system group(Get with OID starting from 1.3.6.1.2.1.1.1. to 1.3.6.1.2.1.1.7)
2. NUT replies SNMPv2C Response with correct OID values to TN

1st Packet(sent by TN)

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	7A	
	version	1(SNMPv2C)	02	01	01

community			public	04	06	70 75 62 6C 69 63
Data	PDU type		GetRequest	A0	6D	
	request-id		12	02	01	0C
	error-status		0	02	01	00
	error-index		0	02	01	00
				30	62	
	variable-bindings			30	0C	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.6.0 (sysLocation.0)	06	08	2b 06 01 02 01 01 06 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.7.0 (sysServices.0)	06	08	2b 06 01 02 01 01 07 00
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01

community			public	04	06	70 75 62 6C 69 63
Data	PDU type		Response	A2	*	
	request-id		12	02	01*	0C
	error-status		0	02	01	00
	error-index		0	02	01	00
				30	*	
	variable-bindings			30	*	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00*
		value	octet string of NUT system description	06	*	variable string*
				30	*	
		name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
		value	Object identifier of NUT system objectID	06	*	variable object identifier*
				30	*	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	Time ticks of NUT system up time	43		variable time ticks*
				30	*	
		name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
		value	octet string of NUT system contact information	06	*	variable string*
				30	*	
		name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
		value	octet string of NUT system name	06	*	variable string*
				30	*	
		name	1.3.6.1.2.1.1.6.0 (sysLocation.0)	06	08	2b 06 01 02 01 01 06 00
		value	octet string of NUT system description	06	*	variable string*
				30	*	
		name	1.3.6.1.2.1.1.7.0 (sysServices.0)	06	08	2b 06 01 02 01 01 07 00
		value	Integer value of	02	01	variable integer

				NUT system services			values*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C Get scalar object request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. value fields for the seven OID in system group are correct values with correct syntax type and within the range field.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.1.5 Get multiple scalar objects with noSuchObject, noSuchInstance

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest multiple object packets with error response from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

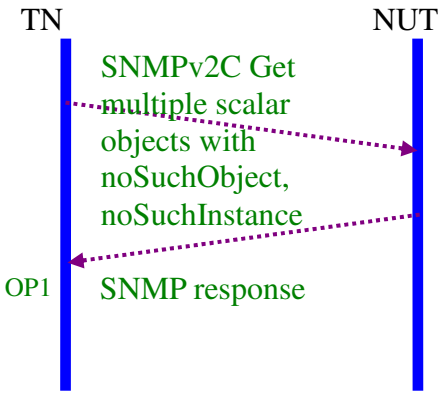
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest scalar object to NUT by issuing SNMPv2C GetRequest to test with error OID.
2. NUT replies SNMPv2C Response with correct OID values to TN. Error OIDs will be responded accordingly.

1st Packet(sent by TN)

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	42	
	version	1(SNMPv2C)	02	01	01

	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	35	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	2A	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.255	06	08	2b 06 01 02 01 01 81 7F
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.2.1 00	06	08	2b 06 01 02 01 01 02 64
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	12
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	0C	
	name	1.3.6.1.2.1.1.255	06	08	2b 06 01 02 01 01 81 7F
	value	noSuchObject	80	00	
			30	*	
	name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
	value	sysUpTime of NUT system	43	*	TimeTicks
			30	1C	

			name	1.3.6.1.2.1.1.2.100	06	08	2b 06 01 02 01 01 02 64
			value	noSuchInstance	81	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C Get scalar object request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the value field of responding variable binding list are noSuchObject, correct sysUpTime value and noSuchInstance

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.2 Get tabular objects
v6SNMPv2C1.1.2.1. Get OIDs from the same table
v6SNMPv2C1.1.2.1.1 Get OIDs from the same table with correct values

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest objects packet from a table(the ifTable in this test scenario) in the MIB issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

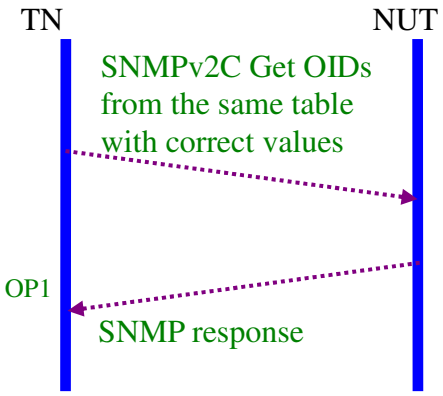
Refer Common Test Setup

Walk Preparation

Walk MIB II IfTable to get correct IfIndex

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest objects to NUT by issuing SNMPv2C Get to get ifIndex and ifType in ifTable
2. NUT replies SNMPv2C Response with correct OID values to TN

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		Any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	Value

				30	*	
version			1(SNMPv2C)	02	01	01
community			public	04	06	70 75 62 6C 69 63
Data	PDU type		GetRequest	A0	*	
	request-id		12	02	01	0C
	error-status		0	02	01	00
	error-index		0	02	01	00
				30	*	
	variable-bindings			30	*	
		name	1.3.6.1.2.1.2.2.1.1.*[ifIndex]* *(1 : ifIndex)*	06	*	2b 06 01 02 01 02 02 01 01 *
		value	NULL	05	00	
				30	*	
		name	1.3.6.1.2.1.2.2.1.3.*[ifIndex]* *(3 : ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
version		1(SNMPv2C)	02	01	01
community		public	04	06	70 75 62 6C 69 63
Data	PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.2.1.1.*[ifIndex]* *(1 : ifIndex)*	06	*	2b 06 01 02 01 02 02 01 01 *
	value	*	02	*	Integer32
			30	*	
	name	1.3.6.1.2.1.2.2.1.3.*[ifIndex]* *(3 : ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
	value	*	02	*	Integer

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1:

TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest tabular objects correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. value field are the correct ifIndex and ifType value with correct syntax type and within the defined range field

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.2.1.2 Get OIDs from the same table with noSuchObject

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest object packet from a table(the ifTable in this test scenario) with noSuchObject issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

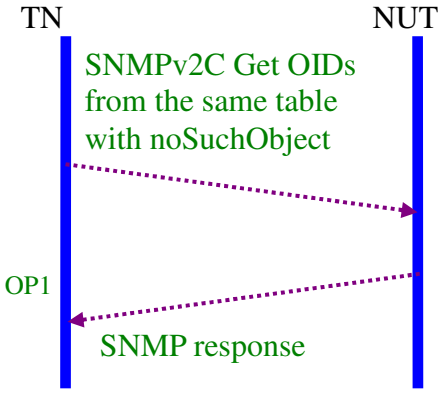
Refer Common Test Setup

Walk Preparation

Walk MIB II IfTable to get correct IfIndex

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest object to NUT by issuing SNMPv2C Get to get ifIndex and ifType and noSuchObject error OID in ifTable.
2. NUT replies SNMPv2C Response with correct values to TN

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		Any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	Value
			30	*	

	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.2.2.1.1.*[ifIndex]* *(1 : ifIndex)*	06	*	2b 06 01 02 01 02 02 01 01 *
			value	NULL	05	00	
					30	0E	
			name	1.3.6.1.2.1.2.2.1.255	06	0A	2b 06 01 02 01 02 02 01 81 7F
			value	NULL	05	00	
					30	*	
			name	1.3.6.1.2.1.2.2.1.3.*[ifIndex]* *(3 : ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP manager (NUT) to SNMP agent (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.2.1.1.*[ifIndex]* *(1 : ifIndex)*	06	*	2b 06 01 02 01 02 02 01 01 *
	value	*	02	*	Integer32
			30	0E	
	name	1.3.6.1.2.1.2.2.1.	06	0A	2b 06 01 02 01 02

				255			02 01 81 7F
			value	noSuchObject	80	00	
					30	*	
			name	1.3.6.1.2.1.2.2.1. 3.*[ifIndex]* *(3 : ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
			value	*	02	*	Integer

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1:

TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest tabular objects and noSuchObject correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. value field is the correct ifIndex and ifType value with correct syntax type and within the defined range field and noSuchObject for the error OID.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.2.1.3 Get OIDs from the same table with noSuchInstance

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest object packet from a table(the ifTable in this test scenario) with noSuchInstance issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

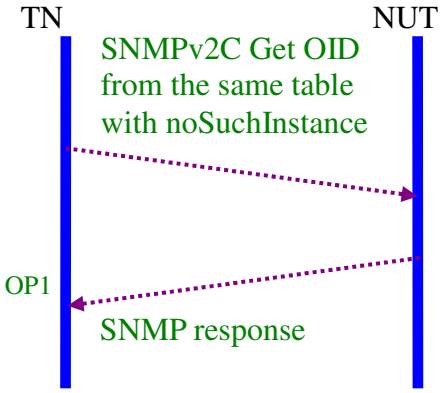
Refer Common Test Setup

Walk Preparation

Walk MIB II IfTable to get correct IfIndex

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest object to NUT by issuing SNMPv2C Get to get ifIndex and ifType and noSuchInstance error OID in ifTable.
2. NUT replies SNMPv2C Response with correct values to TN

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	Value
			30	*	

	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.2.2.1.1.[ifIndex] (1 : ifIndex)	06	*	2b 06 01 02 01 02 02 01 01 *
			value	NULL	05	00	
					30	0E	
			name	1.3.6.1.2.1.2.2.1.3.*[ifIndex=200]* *(3 : ifType)* *Note: NUT must not have interface ID with the value of 200)*	06	0B	2b 06 01 02 01 02 02 01 03 81 48
			value	NULL	05	00	
					30	*	
			name	1.3.6.1.2.1.2.2.1.3.*[ifIndex]* *(3 : ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variab		30	*	
		name	1.3.6.1.2.1.2.2.1.	06	*	2b 06 01 02 01 02

		le-bin din gs		1.*[ifIndex]* *(1 : ifIndex)*			02 01 01 *
			value	*	02	*	Integer32
					30	0E	
			name	1.3.6.1.2.1.2.2.1.3.*[ifIndex=200]* *(3 : ifType)*	06	0B	2b 06 01 02 01 02 02 01 03 81 48
			value	noSuchInstance	81	00	
					30	*	
			name	1.3.6.1.2.1.2.2.1.3.*[ifIndex]* *(3 : ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
			value	*	02	*	Integer

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1:

TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest tabular and error objects correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. value field is the correct ifIndex and ifType value with correct syntax type and with defined range value and noSuchInstance for error OID.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.2.2 Get OIDs from different tables

v6SNMPv2C1.1.2.2.1 Get OIDs from different tables with correct values

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest object packet from different table(the ifTable and udpTable) in the MIB issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

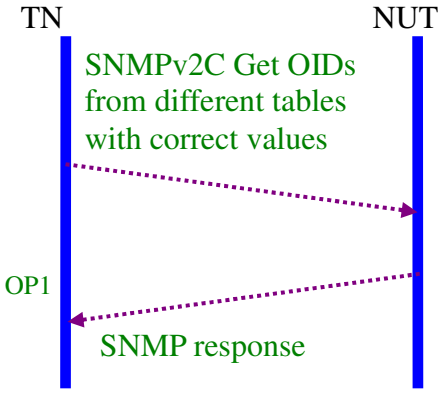
Refer Common Test Setup

Walk Preparation

Walk ifTable and udpTable

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest object to NUT by issuing SNMPv2C Get to get ifType from ifTable and udpLocalAddress in udpTable.
2. NUT replies SNMPv2C Response with correct OID values to TN

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	

	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.2.2.1.3. *[ifIndex]* *(3 :ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
			value	NULL	05	00	
					30	*	
			name	1.3.6.1.2.1.1.7.5.1.1.*[ifIndex]* *(1: udpLocalAddress)*	06	*	2b 06 01 02 01 01 07 05 01 01 *
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	01	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.2.1.3. *[ifIndex]* *(3 :ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
	value	*	02	*	*
			30	*	
	name	1.3.6.1.2.1.1.7.5.1.1.*[ifIndex]* *(1:*	06	*	2b 06 01 02 01 01 07 05 01 01*

				udpLocalAddress)			
			value	IPAddress	40	04	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. correct value field is the tested OID of NUT with value within the range field and syntax type.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.2.2.2 Get OIDs from different tables with noSuchObject

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest object packet from different table(the ifTable and udpTable) in this test scenario issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

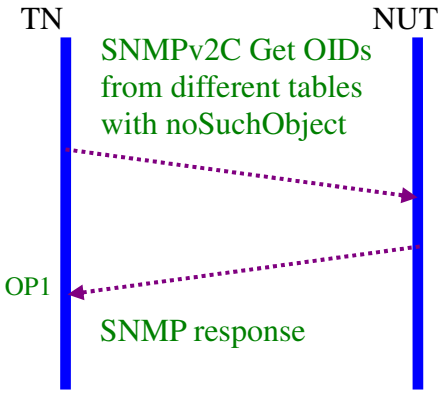
Refer Common Test Setup

Walk Preparation

Walk ifTable and udpTable

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest object to NUT by issuing SNMPv2C Get to get ifType and noSuchObject from ifTable and udpLocalAddress and noSuchObject in udpTable.
2. NUT replies SNMPv2C Response with correct OID values to TN

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	

	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.2.2.1.3. *[ifIndex]* *(3 :ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
			value	NULL	05	00	
			name	1.3.6.1.2.1.2.2.1.255.*[ifIndex]*	06	*	2b 06 01 02 01 02 02 01 81 7F *
			value	NULL	05	00	
					30	*	
			name	1.3.6.1.2.1.1.7.5.1.1. *[udpLocalAddress].[udpLocalPort]*	06	*	2b 06 01 02 01 01 07 05 01 01 *
			value	NULL	05	00	
					30	*	
			name	1.3.6.1.2.1.1.7.5.1.255.*[udpLocalAddress].[udpLocalPort]*	06	*	2b 06 01 02 01 01 07 05 01 81 7F *
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	01	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	

		variable-bindings			30	*	
			name	1.3.6.1.2.1.2.2.1.3 *[ifIndex]* *(3 :ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
			value	*	02	*	*
					30	*	
			name	1.3.6.1.2.1.2.2.1.255.*[ifIndex]*	06	*	2b 06 01 02 01 02 02 01 81 7F *
			value	noSuchObject	80	00	
					30	*	
			name	1.3.6.1.2.1.1.7.5.1.1.*[udpLocalAddress].[udpLocalPort]*	06	*	2b 06 01 02 01 01 07 05 01 01*
			value	*	40	04	*
					30	*	
			name	1.3.6.1.2.1.1.7.5.1.255.*[udpLocalAddress].[udpLocalPort]*	06	*	2b 06 01 02 01 01 07 05 01 81 7F *
			value	noSuchObject	80	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest tabular objects correctly.

Received packet with
1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. value field is the correct OID value of NUT and noSuchObject for error OID.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.1.2.2.3 Get OIDs from different tables with noSuchInstance

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetRequest object packet from different table(the ifTable and udpTable) in this test scenario.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

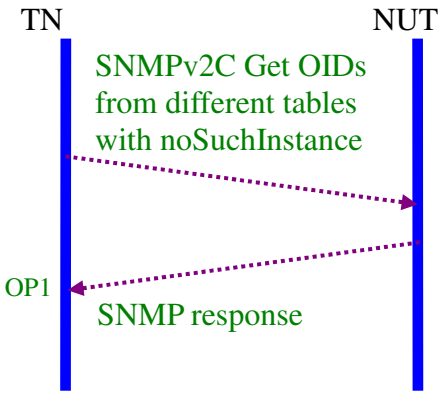
Refer Common Test Setup

Walk Preparation

Walk ifTable and udpTable

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest object to NUT by issuing SNMPv2C Get to get ifType and noSuchInstance from ifTable and to get udpLocalAddress and noSuchInstance in udpTable.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01

community			public	04	06	70 75 62 6C 69 63
Data	PDU type		GetRequest	A0	*	
	request-id		12	02	01	0C
	error-status		0	02	01	00
	error-index		0	02	01	00
				30	*	
	variable-bindings			30	*	
		name	1.3.6.1.2.1.2.2.1.3. *[ifIndex] (3 :ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
		value	NULL	05	00	
				30	0D	
		name	1.3.6.1.2.1.2.2.1.3.*[ifIndex=200] (3 : ifType)* Note: NUT must not have interface ID with value of 200	06	09	2b 06 01 02 01 02 02 01 03 81 48
		value	NULL	05	00	
				30	*	
		name	1.3.6.1.2.1.1.7.5.1.1.*[udpLocalAddress][udpLocalPort]*	06	*	2b 06 01 02 01 01 07 05 01 01 *
		value	NULL	05	00	
				30	0F	
		name	1.3.6.1.2.1.1.7.5.1.1.*[udpLocalAddress].[udpLocalPort]* Note: NUT must not have interface ID with value of 200	06	0B	2b 06 01 02 01 01 07 05 01 01 81 48
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

				30	*	
version			1(SNMPv2C)	02	01	01
community			public	04	06	70 75 62 6C 69 63
Data	PDU type		Response	A2	01	
	request-id		12	02	01	0C
	error-status		0	02	01	00
	error-index		0	02	01	00
				30	*	
	variable-bindings			30	*	
		name	1.3.6.1.2.1.2.2.1. .3. *[ifIndex] (3 :ifType)*	06	*	2b 06 01 02 01 02 02 01 03 *
		value	*	02	*	Integer
				30	0D	
		name	1.3.6.1.2.1.2.2.1. 3.*[ifIndex=200] (3 : ifType)* Note: NUT must not have interface ID with value of 200	06	09	2b 06 01 02 01 02 02 01 03 81 48
		value	noSuchInstance	81	00	
				30	*	
		name	1.3.6.1.2.1.1.7.5. 1.1.*[index=0.0.0 .0.200]*	06	*	2b 06 01 02 01 01 07 05 01 01 00 00 00 00 81 C8
		value	IPAddress	40	04	*
				30	12	
		name	1.3.6.1.2.1.1.7.5. 1.1.*[Index=0.0.0 .0.255]*	06	10	2b 06 01 02 01 01 07 05 01 01 00 00 00 00 81 7F
		value	noSuchInstance	81	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest tabular objects correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community PDU type =A2

2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. correct value field is the tested OID of NUT and error value is noSuchInstance

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.2 Get RequestID Correlation Check

Purpose

Verify that NUT playing the SNMPv2C agent can process each unique requestID from the GetRequest object packet issued by the SNMPv2C manager. Ten SNMP packets with continuous requestID starting from1 will be sent to NUT.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

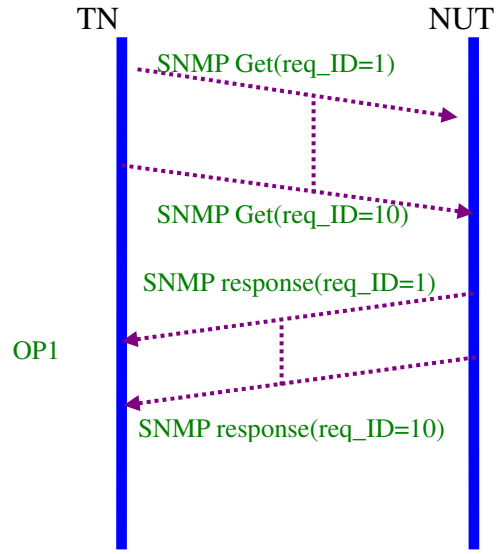
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest scalar object to NUT by issuing SNMPv2C Get with request ID starting from 1 to 10.
2. NUT replies SNMPv2C Response with correct requestID to TN.

Sending packets

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header OP1	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			1(SNMPv2C)	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	19	
		request-id		*	02	***01***	****(value starts from 1)***
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Receiving packets

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			NUT_ADDRESS		
	Destination Address			TN_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	*	
	version			1(SNMPv2C)	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		Response	A2	*	
		request-id		*	02	01	*(from 1 to 10)
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	*	
		variable-bindings			30	*	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	TimeTicks	43	*	TimeTicks

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.

NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Repeat the above test scenarios until the requestID is 10.

Judgment

OP1:
TN received 10 SNMPv2C responses from NUT responding to SNMPv2C GetRequest correctly.

The received 10 packets are with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. The request-id should be 1 to 10 as the previously sent SNMP GetRequests.
3. error-status must be equal to zero and error-index must be equal to zero

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.1

v6SNMPv2C1.3 Error Check

All error tests shall be followed by a SNMPv2C Get sysUpTime to check that the NUT is still functioning normally.

For detailed SNMP packet formats for Get sysUpTime, please refer to those listed in Pre-Test.

v6SNMPv2C1.3.1 Get with sequence_of error
v6SNMPv2C1.3.1.1 Get with sequence_of type error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetRequest with invalid sequence_of type field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

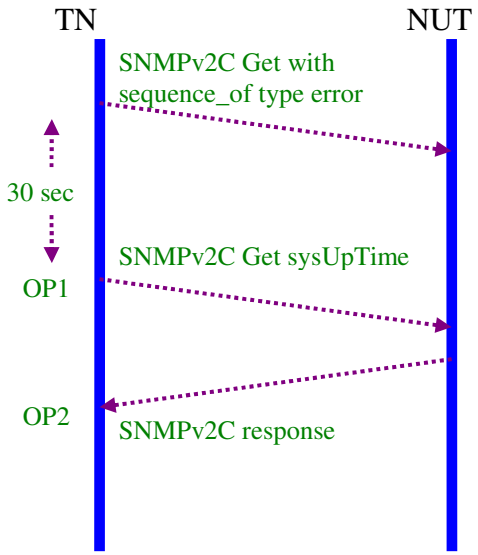
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows:



1. TN sends SNMPv2C GetRequest with sequence_of type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				00	26	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with sequence_of type error packet.
OP2: TN received correct Response with SysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.1.2 Get with sequence_of length error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetRequest with error sequence_of length field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

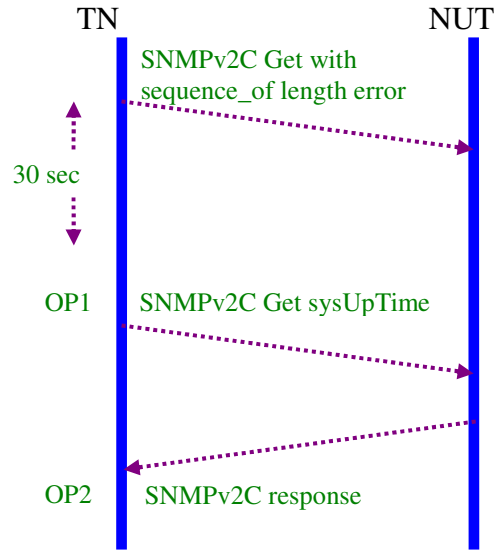
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with sequence_of length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	***0F***	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with sequence_of length error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol Sec 4.2

v6SNMPv2C1.3.2 Get with version number error
v6SNMPv2C1.3.2.1 Get with version number type error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetRequest with invalid version number type field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

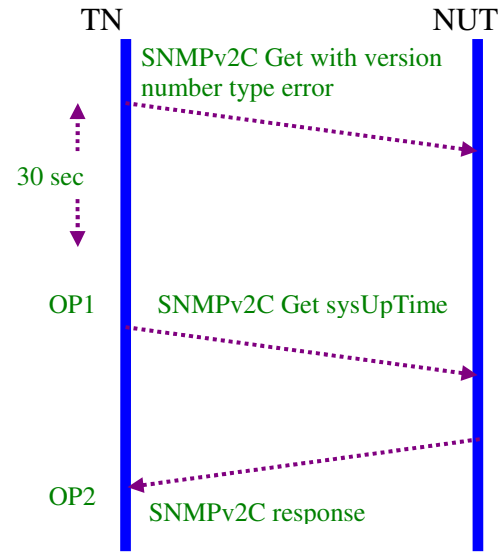
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with version number type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	***00***	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with version number type error packet.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.2.2 Get with version number length error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetRequest with invalid version number length field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

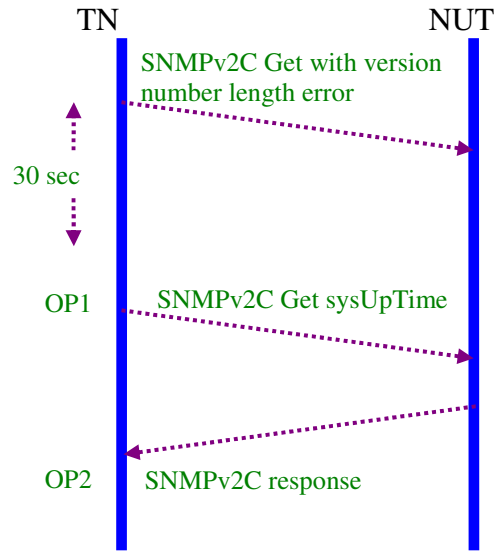
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with version number length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	26	
	version		SNMPv2C	02	***05***	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with version number length error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.2.3 Get with version number value error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetRequest with invalid version number value field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

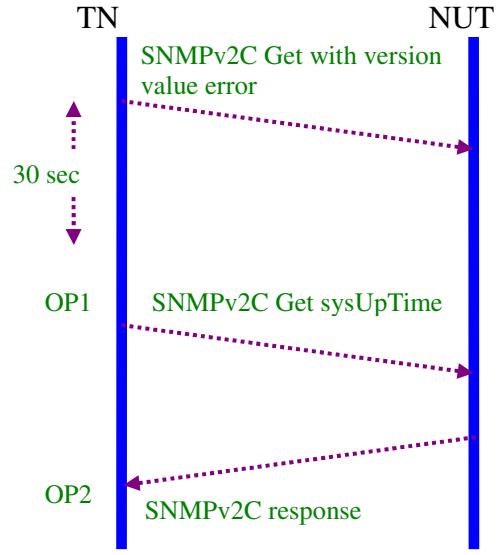
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with version number value error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2C	02	01	***20***
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	19	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with version value error packet.
OP2 TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.3 Get with community error
v6SNMPv2C1.3.3.1 Get with community type error

Purpose

Verify that NUT playing as agent can properly detect the GetRequest with error community type in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

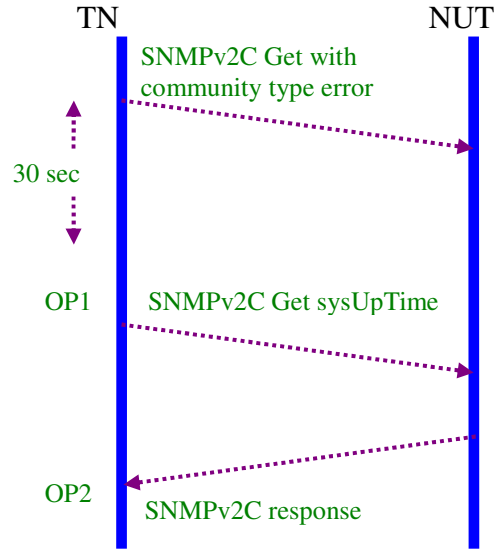
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with community type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN send SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	***02***	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with community type value error packet.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetRequest with community type value error packet as expected.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.3.2 Get with community length error

Purpose

Verify that NUT playing as agent can properly detect the SNMPv2C GetRequest with invalid community length in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

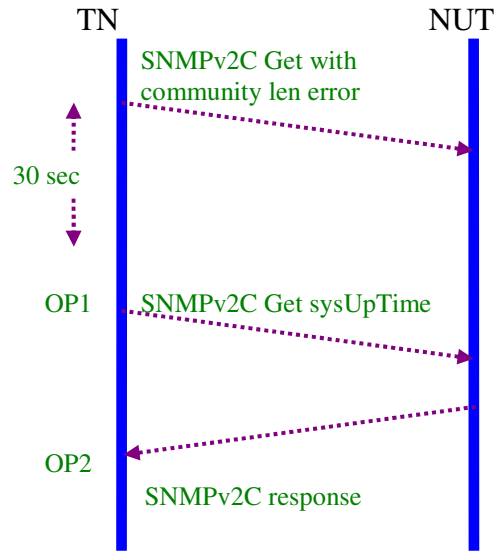
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with community length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			1(SNMPv2C)	02	01	01
	community			public	04	***0F***	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	19	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with community length value error packet.

OP2 TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.3.3 Get with community value error
v6SNMPv2C1.3.3.3.1 Empty community_string

Purpose

Verify that NUT playing as agent can properly detect the SNMPv2C GetRequest with error community value in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

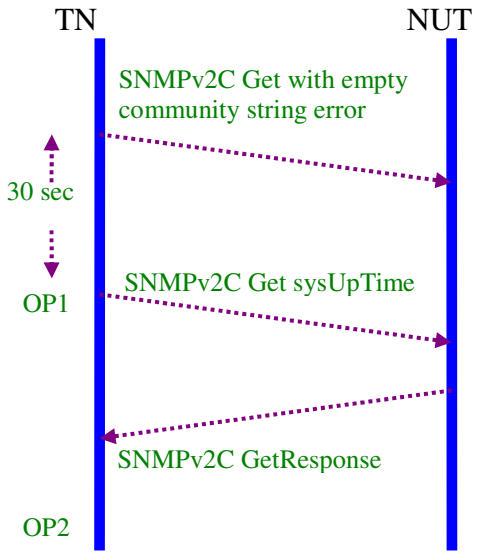
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with empty community string to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			Any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community			***04***	***06***	
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Judgment

OP1: NUT will silently discard this SNMPv2C GetRequest with empty community string value error packet.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.3.3.2 Inconsistent community_string

Purpose

Verify that NUT playing as agent can properly detect the GetRequest with inconsistent community string in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

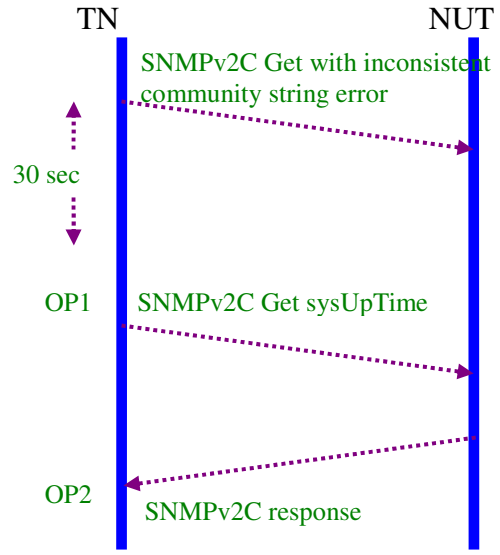
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with inconsistent community string to NUT.
2. NUT discards the datagram and continues to respond to normal requests
3. TN send SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		*pupuic* (public)	04	06	70 75 ***70 75*** 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Judgment

OP1: NUT will silently discard this SNMPv2C GetRequest with inconsistent community string packet.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.3.3.3 community_string with CarriageReturn LineFeed

Purpose

Verify that NUT playing as agent can properly detect the SNMPv2C GetRequest with CR and LF error community value in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

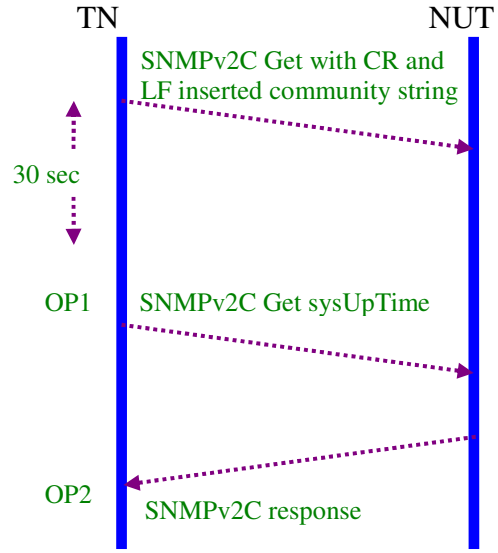
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with CR and LF inserted in the community string error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		Public	04	06	70 75 ***0d 0a*** 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with CR and LF inserted community string packet.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.4. Get with PDU error

v6SNMPv2C1.3.4.1 Get with PDU type error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetRequest with invalid PDU type field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

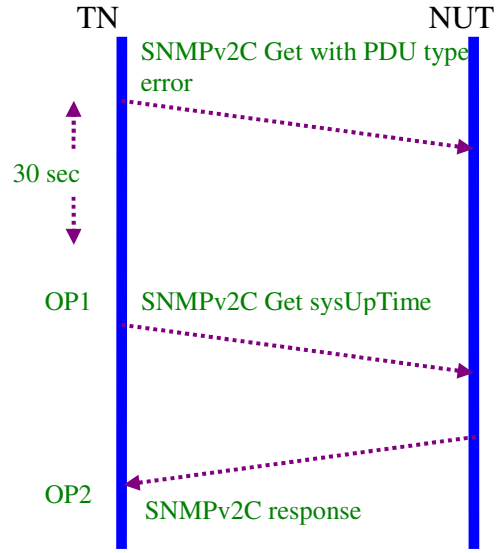
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with PDU type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	***0A***	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 sysUpTime.0	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with PDU type error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.4.2 Get with PDU length error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetRequest with invalid PDU length field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

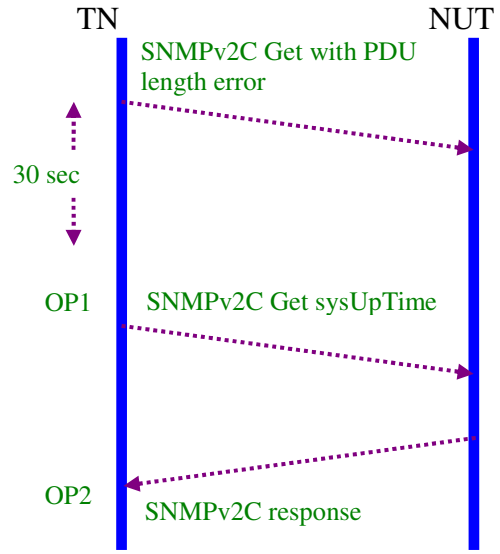
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with PDU length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26*	
	version			SNMPv2C	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	***00***	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid PDU length error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.5 Get with request ID error

v6SNMPv2C1.3.5.1 Get with request ID type error

Purpose

Verify that SNMPv2C agent can properly detect the SNMPv2C GetRequest with request ID field type error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

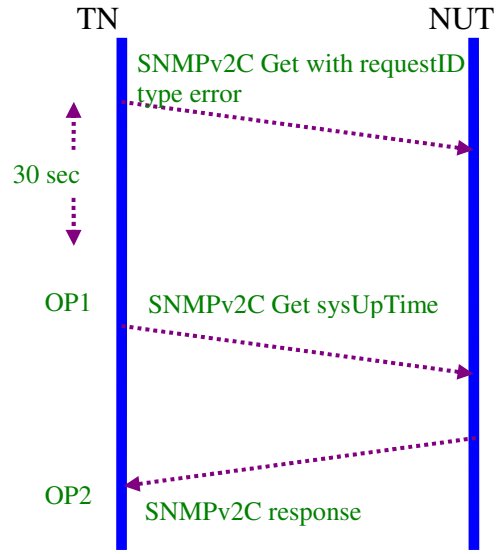
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with request ID type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	19	
		request-id		12	***20***	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid requestID type error packet.

Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetRequest with invalid requestID type error packet.

OP2: TN received correct Response with SysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.5.2 Get with request ID length error

Purpose

Verify that SNMPv2C agent can properly detect the SNMPv2C GetRequest with request ID field len error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

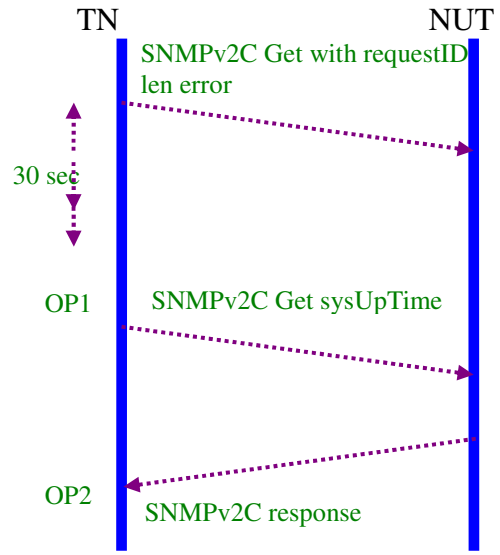
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with request ID length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2C	02	01	01
	community			pubic	04	06	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	19	
		request-id		12	02	***0F***	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 sysUpTime.0	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid requestID length error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.5.3 Get with request ID value error
v6SNMPv2C1.3.5.3.1 Get with requestID greater than maximum value(214783647,0x0CCD569F)

Purpose

Verify that SNMPv2C agent can properly detect the GetRequest with request ID value exceeding the maximum possible value in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

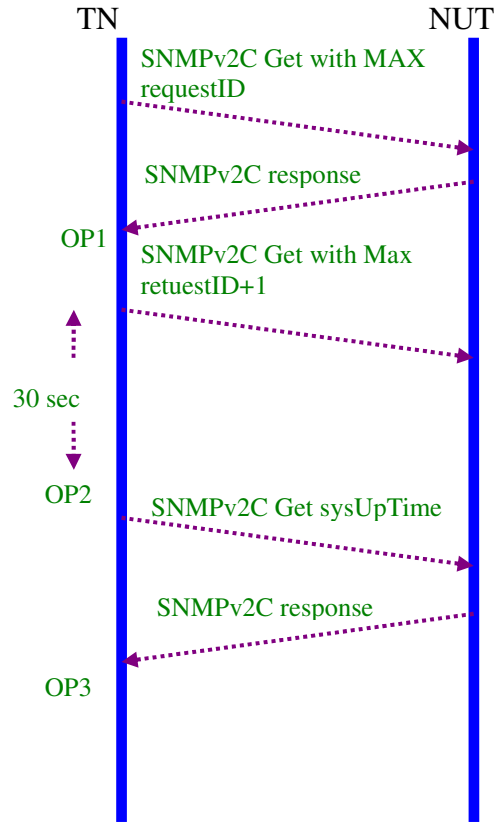
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with requestID value =214783647 to NUT.
2. NUT returns Response.
3. TN sends SNMPv2C GetRequest with requestID value =214783648 to NUT.
4. NUT discards the datagram and continues to respond to normal requests.
5. TN send SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent

is alive

6. NUT return correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	29	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	1C	
		request-id	214783647	02	***04***	***0C CD 56 9F***
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			NUT_ADDRESS		
	Destination Address			TN_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	*	
		request-id	214783647	02	***04***	***0C CD 56 9F***
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		var		30	*	

		iab le-bin din gs	name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	TimeTicks of NUT system up time	43	*	TimeTicks

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	29	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	1C	
		request-id	214783648	02	***04***	***0C CD 56 A0***
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		var iab le-bin din gs		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT returns normal Response with correct sysUpTime
OP2: NUT will silently discard this SNMPv2C GetRequest with request ID exceeding the maximum value.
Note : Warning will be the test judgement when NUT does not discard this SNMPv2C GetRequest with request ID exceeding the maximum value error packet as expected.
OP3: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management

Protocol, Sec 3

v6SNMPv2C1.3.5.3.2 Get with requestID smaller than minimum value(-214783648,0xF332A960)

Purpose

Verify that SNMPv2C agent can properly detect the GetRequest with request ID field below the minimum value in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

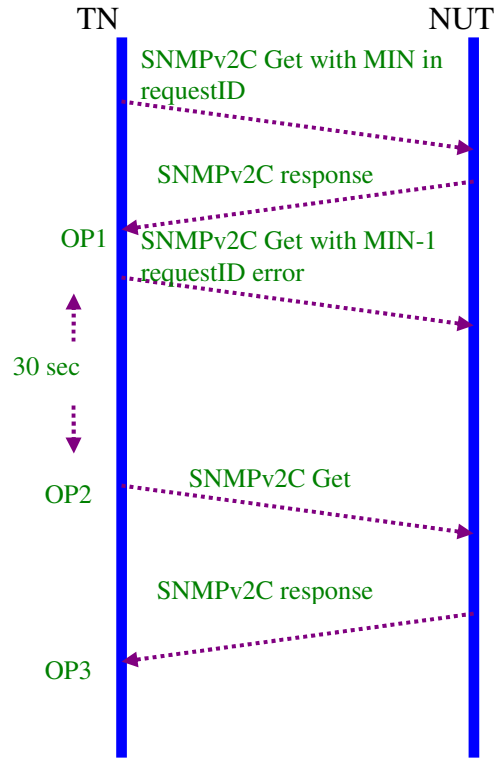
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with requestID value =-214783648 to NUT.
2. NUT returns with Response.
3. TN sends SNMPv2C GetRequest with requestID value =-214783649 to NUT.
4. NUT discards the datagram and continues to respond to normal requests.
5. TN send SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.

6. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)							
IP Header	Source Address			TN_ADDRESS			
	Destination Address			NUT_ADDRESS			
UDP Header	Source Port			any			
	Destination Port			161			
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)			
				type	len	value	
				30	29		
	version		1(SNMPv2C)	02	01	01	
	community		public	04	06	70 75 62 6C 69 63	
	Data	PDU type	GetRequest	A1	1C		
		request-id	-214783648	02	***04***	***F3 32 A9 60***	
		error-status	0	02	01	00	
		error-index	0	02	01	00	
				30	0E		
		variable-bindings		30	0C		
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address			NUT_ADDRESS			
	Destination Address			TN_ADDRESS			
UDP Header	Source Port			161			
	Destination Port			Same as the source port in 1st packet			
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)			
				type	len	value	
				30	*		
	version		1(SNMPv2C)	02	01	01	
	community		public	04	06	70 75 62 6C 69 63	
	Data	PDU type	Response	A2	*		
		request-id	-214783648	02	***04***	***F3 32 A9 60***	
		error-status	0	02	01	00	
		error-index	0	02	01	00	
				30	*		
		variab		30	*		
			name	1.3.6.1.2.1.1.3.0	06	08	2b 06 01 02 01 01

		le-bindings		(sysUpTime.0)			03 00
			value	NUT system up-time	43	*	TimeTicks

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)							
IP Header	Source Address			TN_ADDRESS			
	Destination Address			NUT_ADDRESS			
UDP Header	Source Port			any			
	Destination Port			161			
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)			
				type	len	value	
				30	29		
	version		SNMPv2C	02	01	01	
	community		public	04	06	70 75 62 6C 69 63	
	Data	PDU type	GetRequest	A0	1C		
		request-id	-214783649	02	***04***	***F3 32 A9 5A***	
		error-status	0	02	01	00	
		error-index	0	02	01	00	
				30	0E		
		variable-bindings		30	0C		
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT returns Response with correct sysUpTime
OP2: NUT will silently discard this malformed SNMPv2C GetRequest with requestID exceeding minimum value.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetRequest with requestID exceeding minimum value error packet as expected.
OP3: TN received correct Response with SysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 3

v6SNMPv2C1.3.6 Get with error-status error
v6SNMPv2C1.3.6.1 Get with error-status type error

Purpose

Verify that NUT playing as agent can properly detect the SNMPv2C GetRequest with error-status type error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

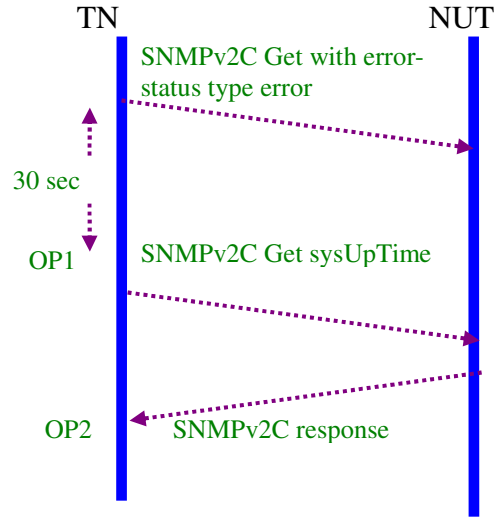
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with error-status type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2C	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	19	
		request-id		12	02	01	0C
		error-status		0	***20***	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid error status type packet.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetRequest with invalid error status type error packet as expected.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.6.2 Get with error-status length error

Purpose

Verify that NUT playing as agent can properly detect the SNMPv2C GetRequest with error-status length error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

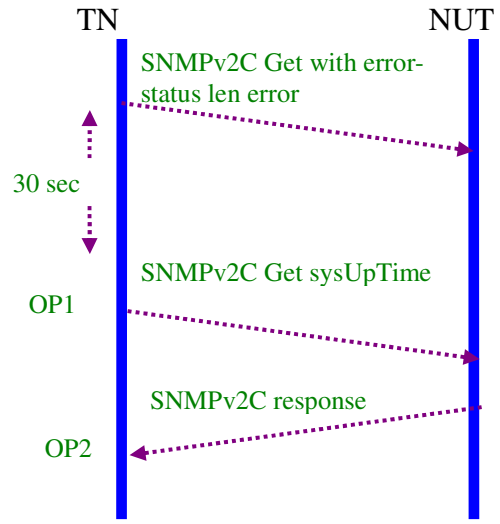
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with error-status length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	***10***	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid error status len error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.6.3 Get with error-status non-zero error

Purpose

Verify that NUT playing as agent can properly detect the SNMPv2C GetRequest with error-status value none zero error in the received packet from SNMPv2C manager and will ignore the error-status and respond with correct value.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

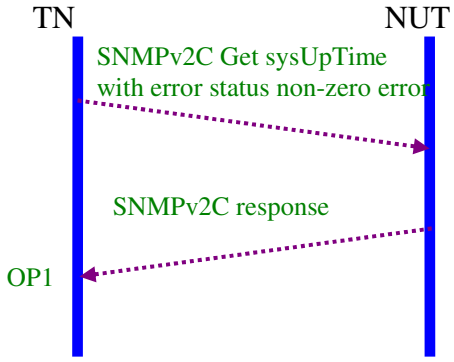
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with non-zero error-status error to NUT.
2. NUT respond correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	26	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63

	D a t a	PDU type		GetRequest	A0	19	
		request-id		12	02	01	0C
		error-status		16	02	01	***10***
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
	value	NUT system up-time	43	*	TimeTicks

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.7 Get with error-index error
v6SNMPv2C1.3.7.1 Get with error-index type error

Purpose

Verify that NUT playing as agent can properly detect invalid SNMPv2C GetRequest with error-index error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

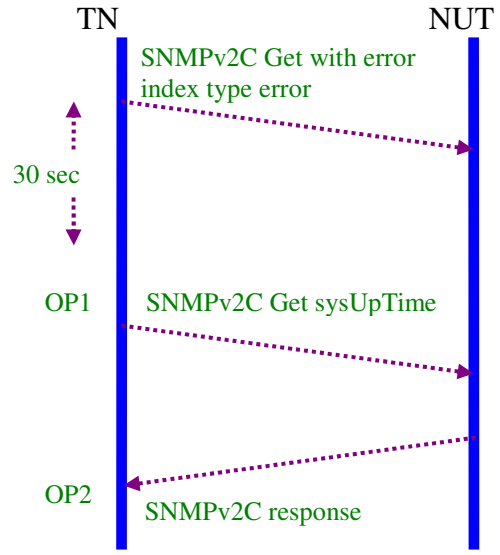
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with error-index type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	***20***	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid error-index type packet.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetRequest with invalid error-index type error packet as expected.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.7.2 Get with error-index length error

Purpose

Verify that NUT playing as agent can properly detect invalid SNMPv2C GetRequest with error-index length error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

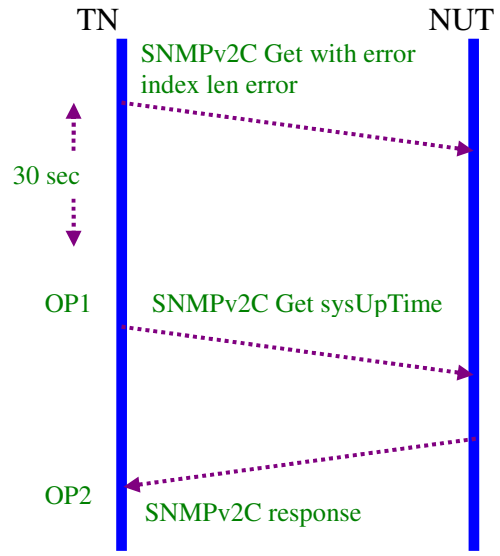
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with error index length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		pubic	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	***10***	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	*	
		request-id	12	02	01	OC
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NUT system up-time	43	*	TimeTicks

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address

TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid error index length packet.
OP2: TN received correct Response with the sysUpTime value

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.7.3 Get with error-index non-zero error

Purpose

Verify that NUT playing as agent can properly detect invalid SNMPv2C GetRequest with error-index non-zero error in the received packet from SNMPv2C manager and will return with correct sysUpTime.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

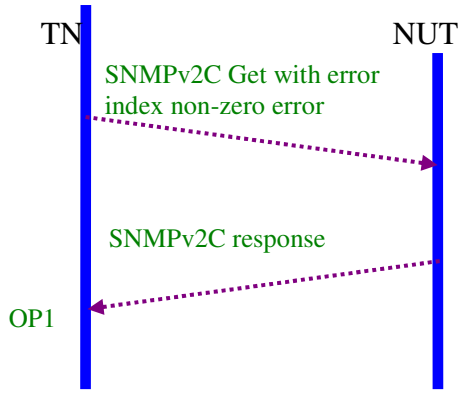
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with error-index non-zero error to NUT.
2. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address		TN_ADDRESS			
	Destination Address		NUT_ADDRESS			
UDP Header	Source Port		any			
	Destination Port		161			
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)			
			type	len	value	
			30	26		
	version	SNMPv2C	02	01	01	
	community	public	04	06	70 75 62 6C 69 63	
	Data PDU type	GetRequest	A0	19		
	Data request-id	12	02	01	0C	

	t a	error-status		0	02	01	00
		error-index		16	02	01	***10***
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received correct Response with the sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.8 Get with variable-bindings error
v6SNMPv2C1.3.8.1 Get with OID type error

Purpose

Verify that SNMPv2C agent can properly detect the SNMPv2C GetRequest with OID encoding type error of variable-binding's name in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

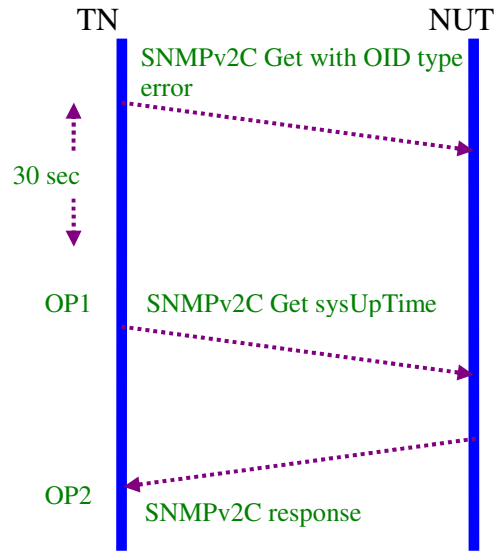
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with OID type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	***07***	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid OID type packet.
OP2: TN received correct Response with the current sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.8.2 Get with OID length error

Purpose

Verify that NUT playing as SNMPv2C agent can properly detect the SNMPv2C GetRequest with OID encoding length error of variable binding's name in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

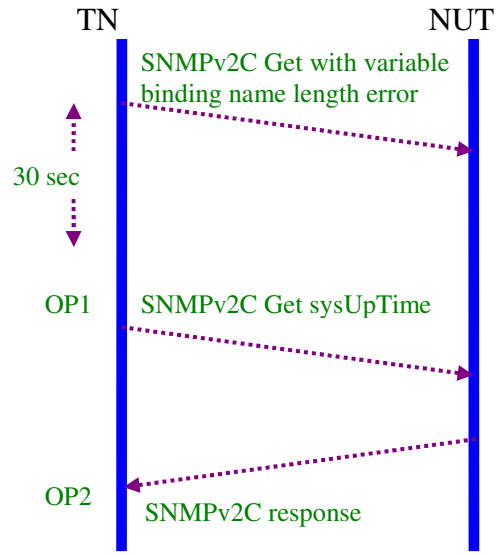
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with OID length value error to NUT.
2. NUT discards the datagram and continue to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2C	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	19	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	***7F***	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with invalid variable name length packet.
OP2 TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C1.3.8.3 Get with OID value error
v6SNMPv2C1.3.8.3.1 Get with FF value in variable-binding's name

Purpose

Verify that NUT playing as SNMPv2C agent can properly detect the SNMPv2C GetRequest with OID coding error in the received packet from SNMPv2C manager and will discard this datagram and continue to respond to normal requests.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

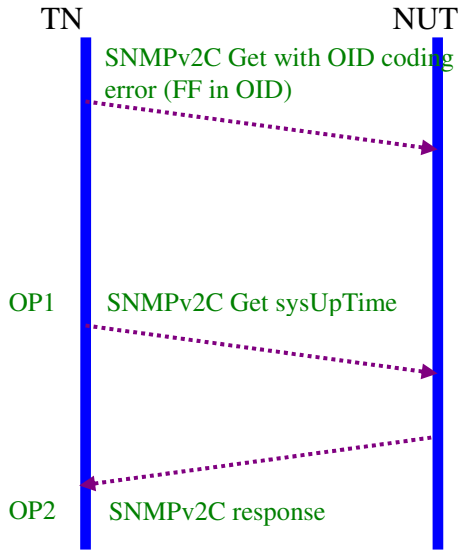
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with OID coding error to NUT.
2. NUT silently discard this datagram and continue to respond to normal requests.
3. TN sends SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	01	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.3.0 sysUpTime.0	06	08	2b 06 01 02 01 01 03 ***FF***
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with OID coding error.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

v6SNMPv2C1.3.8.3.2 Get with variable-binding's value without NULL

Purpose

Verify that SNMPv2C agent can properly detect the GetRequest with varBinding without NULL error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

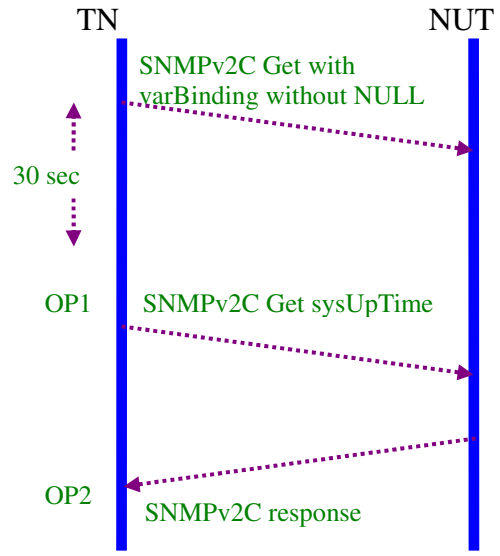
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with variable binding without NULL value error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN send SNMPv2C GetRequest sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	24	
	version			1(SNMPv2C)	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetRequest	A0	17	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0C	
		variable-bindings			30	0A	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime)	06	08	2b 06 01 02 01 01 03 00
			value				

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with variable binding's value without NULL packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol , Sec 4.2

v6SNMPv2C1.3.8.3.3 Get with zero variable-bindings

Purpose

Verify that SNMPv2C agent can properly detect the SNMPv2C GetRequest with zero variable-bindings in the received packet from SNMPv2C manager and will respond with Response with empty variable binding packet.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

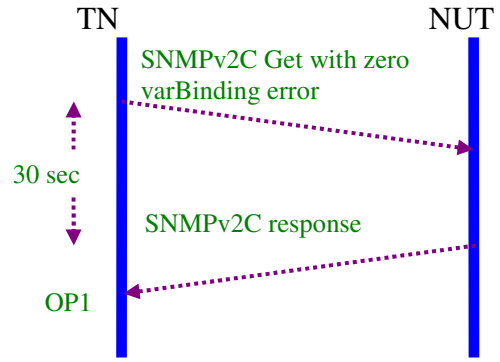
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with zero variable binding error to NUT.
2. NUT returns with Response with empty variable binding.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	D PDU type	GetRequest	A0	0B	

	a t a	request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	00	
		variable-bindings				
			name			
			value			

2nd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
		name			
		value			

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will reply with Response with empty variable binding

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol , Sec 4.2

v6SNMPv2C1.3.8.3.4 128 sub_identifiers check

Purpose

Verify that SNMPv2C agent can properly handle SNMPv2C GetRequest with 128 sub-identifiers in the received packet from SNMPv2C manager and will respond noSuchObject.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

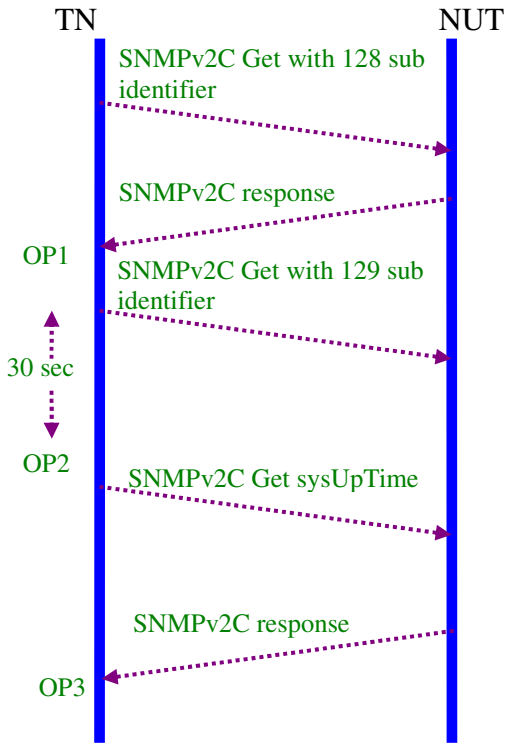
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with 128 sub-identifiers error to NUT.
2. NUT responds with noSuchObject
3. TN sends SNMPv2C GetRequest with 129 sub-identifiers error to NUT.
4. NUT discards the datagram and continues to respond to normal requests.
5. TN sends SNMPv2C GetRequest sysUpTime to NUT for verifying NUT is still alive
6. NUT returns current sysUpTime value

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	81a2 (162)	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	8194 (148)	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	8188 (136)	
		variable-bindings		30	8185 (133)	
		name	1.3.6.1.2.3.4.5.6.7.8…..124.128 (128 sub-IDs)	06	81 80(128)	2b 06 01 02 03 04 05 06 07 …..7C 81 00
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	81a2 (162)	
	version	SNMPv2C	02	01	01

	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		Response	A2	8194 (148)	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	8188 (136)	
		variable-bindings			30	8185 (136)	
			name	1.3.6.1.2.3.4.5.6.7.8…..124.128 (128 sub-IDs)	06	81 80	2b 06 01 02 03 04 05 06 07 …..7C 81 00
			value	***noSuchObject***	***81***	***00***	

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	81a3 (163)	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	GetRequest	A0	8195 (149)	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	8189	
	variable-bindings		30	8186	
	name	1.3.6.1.2.3.4.5.6.7.8…..126 (129 sub-IDs)	06	81 81	2b 06 01 02 03 04 05 06 07 …..7C 7D 81 00
	value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will respond with noSuchObject.
OP2: NUT silently discards this malformed SNMPv2C GetRequest with 129 sub-identifiers packet
OP3: NUT received the correct sysUpTime value

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec. 4.1
RFC 2578, Structure of Management Information Version 2 (SMIv2), Sec 3.5

v6SNMPv2C1.3.9 Get with tooBig message

Purpose

Verify that SNMPv2C agent can properly handle the SNMPv2C GetRequest datagram in the received packet from SNMPv2C manager and will respond either Response when the SNMPv2C agent can reply normally or with tooBig error code when the size of the resultant message is less than or equal to both a local constraint and the maximize size of the SNMPv2C manager.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

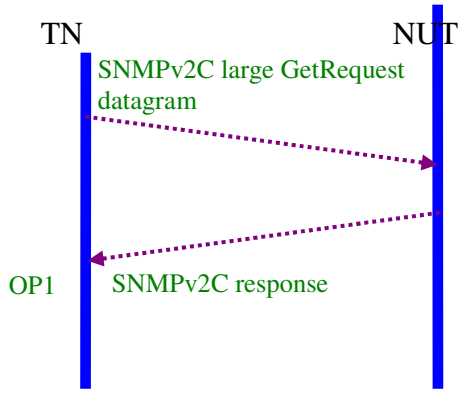
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetRequest with large variable-bindings to NUT.
2. NUT responds with either normal Response or with tooBig.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	82 05 94(1	

					428)	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetRequest	A0	82 05 85(1413)	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	***82 05 78(1400)***	***01 a4(for 100 variable-bindings)***
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	
				30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	
			name			Until 100 repetitions
			value			

2nd Packet is either

Standard query from SNMP manager (NUT) to SNMP agent (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in the 1st packet		
	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	1	02	01	01
	error-index	0	02	01	00
			30	*	
			30	*	
	name	1.3.6.1.2.1.3.0 sysUpTime.0	06	08	2b 06 01 02 01 01 03 00

			value	TimeTicks	43	*	*
					30	0C	
			name	1.3.6.1.2.1.3.0 sysUpTime.0	06	08	2b 06 01 02 01 01 03 00
			value	TimeTicks	43	*	*
							Repeat until 100 repetitions

Or 2nd Packet with tooBig error-status code

Standard query from SNMP manager (NUT) to SNMP agent (TN)						
IP Header	Source Address			NUT_ADDRESS		
	Destination Address			TN_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in the 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	18	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	0B	
		request-id	12	02	01	0C
		error-status	1	02	01	01(tooBig)
		error-index	0	02	01	00
				30	00	
		name				
		value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will respond either with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status and error-index must be equal zero
4. the variable binding list is the 100 variable-bindings in the first GetRequest packet.

Or OP1: NUT will respond with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type

= A2

2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to one(tooBig) and error-index must be equal zero
4. empty variable binding's field.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.1

Group 2 IPv6 SNMPv2C GetNextRequest

Scope
The following tests verify the GetNextRequest commands in IPv6 SNMPv2C protocols.

Overview
Tests in this group verify that a SNMPv2C agent can properly perform the lexicographic ordering correctly and generate the correct SNMPv2C messages with Response PDU according to the SNMPv2C GetNext commands received. These tests also verify a SNMPv2C agent will transmit the appropriate SNMPv2C parameter problem error messages in response to invalid or unknown fields in the received SNMPv2C packets. Make sure the GetNext OID is a valid OID before conducting this GetNextRequest.

v6SNMPv2C2.1 GetNext Operations
v6SNMPv2C2.1.1 GetNext scalar object
v6SNMPv2C2.1.1.1 GetNext single scalar object

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest object packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

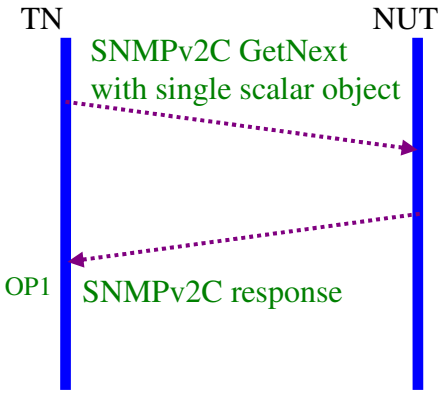
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest single scalar object to NUT by issuing SNMPv2C GetNextRequest to get the next OID after sysDescr 1.3.6.1.2.1.1.1 in system group in MIB II.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

				30	25	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	18	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0D	
		variable-bindings		30	0B	
			name: 1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value: NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
	value	sysDescr of NUT system	04	*	variable string*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNext scalar object request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the first variable binding's name is 1.3.6.1.2.1.1.1.0(sysDescr.0) and its value field is correct syntax type(Octet String).

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.1.2 GetNext single scalar object from non-existent object

Purpose

Verify that NUT playing the SNMPv2C agent can properly handle the GetNextRequest non-existent object packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

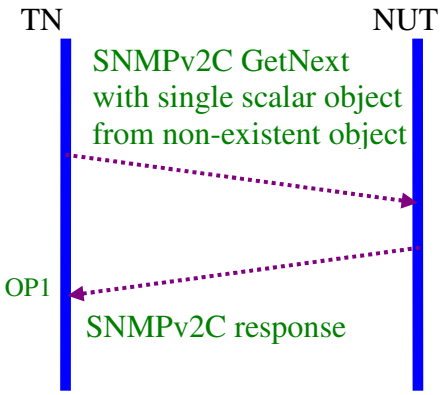
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest scalar object to NUT by issuing SNMPv2C GetNextRequest with non-existent object
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	25	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	D PDU type	GetNextRequest	A1	18	

	a t a	request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0D	
		variable-bindings			30	0B	
			name	1.3.6.1.2.1.1.100	06	07	2b 06 01 02 01 01 64
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.1.0 (ifNumber.0)	06	08	2b 06 01 02 01 02 01 00
	value	Variable integer value of ifNumber.0	02	*	Integer

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNext scalar object request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the first variable binding's name is 1.3.6.1.2.1.2.1.0 (ifNumber.0) and its value field is correct syntax type(Integer32).

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.1.3 GetNext single scalar object from existent instance

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest object packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

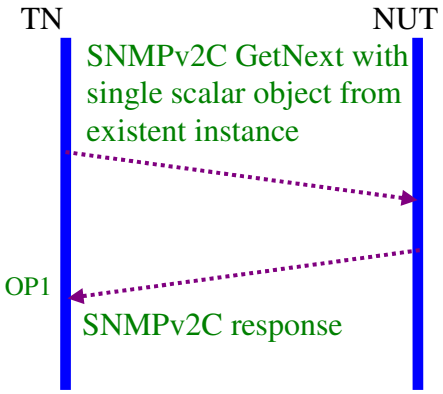
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest scalar object to NUT by issuing SNMPv2C GetNextRequest with existent instance.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	26	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	D PDU type	GetNextRequest	A1	19	

	a t a	request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
	value	Object identifier	06	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNext scalar object request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the first variable binding's name is 1.3.6.1.2.1.1.2.0(sysObjectID.0) and its value field is correct syntax type.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.1.4 GetNext single scalar object from non-existent instance

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest object packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

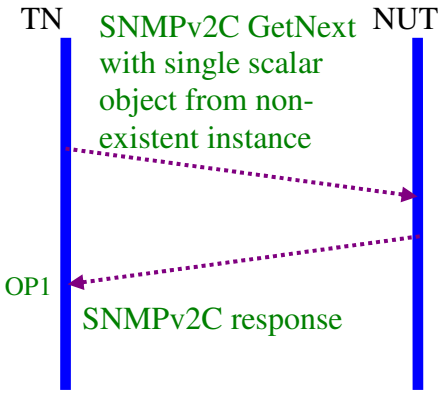
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest scalar object to NUT by issuing SNMPv2C GetNextRequest with non-existent instance.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	26	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	D PDU type	GetNextRequest	A1	19	

	a t a	request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.1.100	06	08	2b 06 01 02 01 01 01 64
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.1.2.0	06	08	2b 06 01 02 01 01 02 00
	value	sysObjectID.0	06	*	Object identifier

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNext scalar object request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the first variable binding's name is 1.3.6.1.2.1.1.2.0(sysObjectID.0) and its value field is correct syntax type.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.1.5 GetNext from 2.0 (endOfMIBView)

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest from 2.0 and return endOfMIBView.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

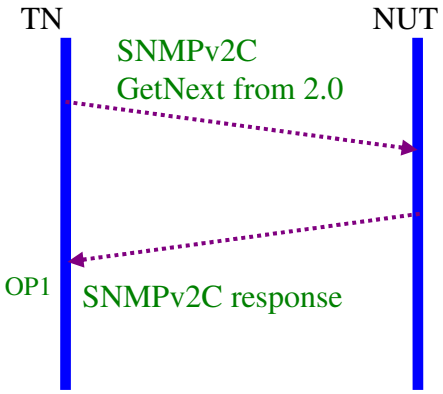
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN send SNMPv2C GetNextRequest scalar object to NUT by issuing SNMPv2C GetNextRequest with 2.0.
2. NUT replies SNMPv2C Response with endOfMIBView values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	19	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	D PDU type	GetNextRequest	A1	12	

	a t a	request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	07	
		variable-bindings			30	05	
			name	2.0	06	01	50
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	1F	
	version	1(SNMPv2C	02	01	01
	community	Public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	12	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	07	
	variable-bindings		30	05	
	name	2.0	06	01	50
	value	endOfMIBView	82	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNext request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. value field is the endOfMIBView

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.1.6 GetNext multiple scalar objects

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest multiple objects packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

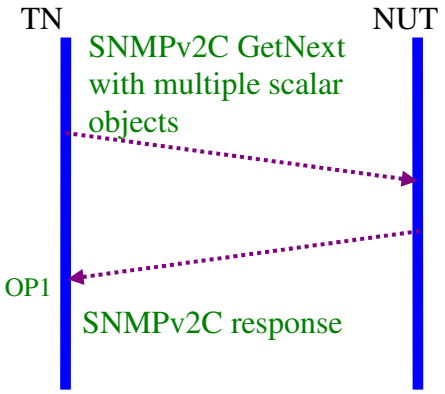
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest multiple scalar objects to NUT by issuing SNMPv2C GetNextRequest to get all scalar OIDs in system group(GetNextRequest with OID starting from 1.3.6.1.2.1.1.1.0 to 1.3.6.1.2.1.1.6.0).
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	6C	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63

	D a t a	PDU type		GetNextRequest	A1	5F	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	54	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	06 08 2b 06 01 02 01 01
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.6.0 (sysLocation.0)	06	08	2b 06 01 02 01 01 06 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	01	
	Data request-id	12	02	01	0C
	Data error-status	0	02	01	00
	Data error-index	0	02	01	00

					30	*	
					30	*	
			name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
			value	Object identifier of NUT system objectID	06	*	variable object identifier*
					30	*	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	Time ticks of NUT system up time	43	*	variable time ticks*
					30	*	
			name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value	octet string of NUT system contact information	04	*	variable string*
					30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01
			value	octet string of NUT system name			variable string*
					30	*	
			name	1.3.6.1.2.1.1.6.0 (sysLocation.0)	06	08	2b 06 01 02 01 01 06 00
			value	octet string of NUT system location			variable string*
					30	*	
			name	1.3.6.1.2.1.1.7.0 (sysServices.0)	06	08	2b 06 01 02 01 01 07 00
			value	Integer value of NUT system services			variable integer values*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN Received SNMPv2C response from NUT responding to SNMPv2C GetNext scalar multiple objects request correctly.

Received Packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. The variable binding's name are 1.3.6.1.2.1.1.2.0(sysObjectID.0), 1.3.6.1.2.1.1.3.0(sysUpTime.0), 1.3.6.1.2.1.1.4.0(sysContact.0), 1.3.6.1.2.1.1.5.0(sysName.0), 1.3.6.1.2.1.1.6.0(sysLocation.0) and 1.3.6.1.2.1.1.7.0(sysServices.0) respectively and their value should be with correct syntax type and within their defined value range .

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.2. GetNext tabular objects
v6SNMPv2C2.1.2.1 GetNext from ifTable

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest object packet from a table(the ifTable in this test scenario) in the MIB issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

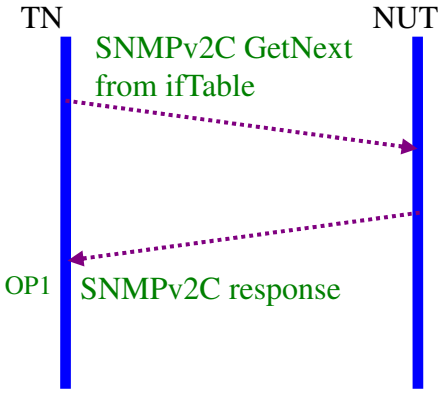
Refer Common Test Setup

Walk Preparation

Walk MIB II IfTable to GetNext correct IfIndex

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest object to NUT by issuing SNMPv2C GetNextRequest to get ifTable.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	25	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	18	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0D	
		variable-bindings		30	0B	
			name	1.3.6.1.2.1.2.2 (ifTable)	06	07	2b 06 01 02 01 02 02
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	OC
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.2.1.1.[ifIndex]	06	*	2b 06 01 02 01 02 02 01 01 *
	value	Integer32	02	*	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNext from ifTable correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned object value is the first index of ifTable

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.2.2 GetNext from ifEntry

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest from ifEntry issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

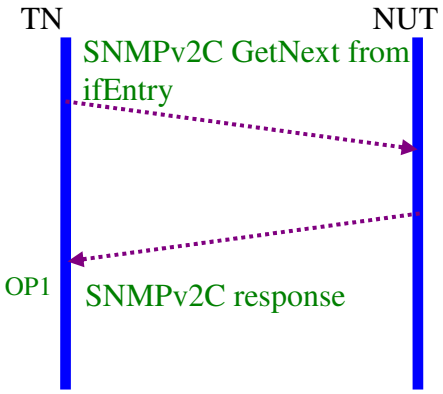
Refer Common Test Setup

Walk Preparation

Walk MIB II IfTable to GetNext correct IfIndex

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest object to NUT by issuing SNMPv2C GetNextRequest to get ifEntry
2. NUT replies SNMPv2C Response with correct OID values to TN

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	26	
	version	1(SNMPv2C)	02	01	01

	community		Public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.2.2.1 (ifEntry)	06	08	2b 06 01 02 01 02 02 01
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	OC
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.2.1.1.[ifIndex]	06	*	2b 06 01 02 01 02 02 01 01 *
	value	Integer32	02	*	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNextRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned object value is the first index of ifTable

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.2.3 GetNext from ifIndex

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest from ifIndex issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

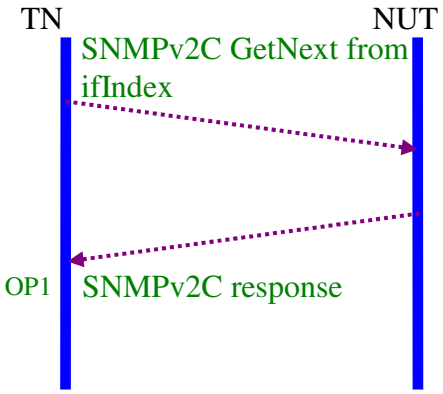
Refer Common Test Setup

Walk Preparation

Walk MIB II IfTable to GetNext correct IfIndex

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest object to NUT by issuing SNMPv2C GetNextRequest to get ifIndex.
2. NUT replies SNMPv2C Response with correct values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	27	
	version	1(SNMPv2C)	02	01	01

	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	1A	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0F	
		variable-bindings		30	0D	
			name	1.3.6.1.2.1.2.2.1.1 (ifIndex)	06	09	2b 06 01 02 01 02 02 01 01
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	OC
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.2.1.1.[ifIndex]	06	*	2b 06 01 02 01 02 02 01 01 *
	value	Integer32	02	*	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNextRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned object value is the first index of ifTable

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.2.4 GetNext from ifIndex.0

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest from ifIndex.0 issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

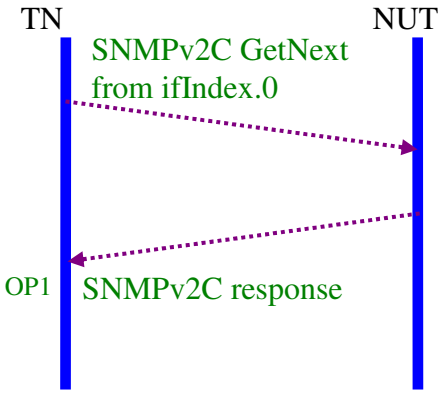
Refer Common Test Setup

Walk Preparation

Walk MIB II IfTable to GetNext correct IfIndex

Procedure

The test sequence is as follows



1. TN send SNMPv2C GetNextRequest object to NUT by issuing SNMPv2C GetNextRequest to get ifIndex.0
2. NUT reply SNMPv2C Response with correct OID values to TN

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	28	
	version	1(SNMPv2C)	02	01	01

	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	1B	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	10	
		variable-bindings		30	0E	
			name	1.3.6.1.2.1.2.2.1.1.0 (ifIndex.0)	06	0A	2b 06 01 02 01 02 02 01 01 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	OC
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.2.1.1.[ifIndex]	06	*	2b 06 01 02 01 02 02 01 01 *
	value	Integer32	02	*	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNextRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned object value is the first index of ifTable

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.2.5 GetNext from ifIndex.10000

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNext ifIndex.10000 issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

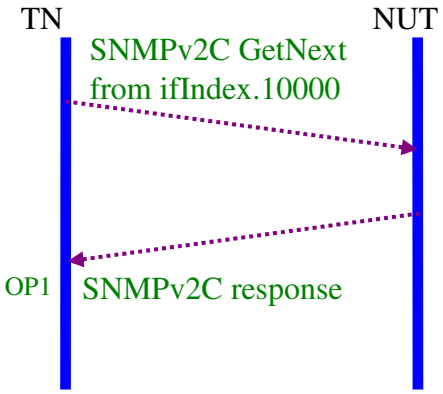
Refer Common Test Setup

Walk Preparation

Walk MIB II IfTable to GetNext correct ifIndex

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest object to NUT by issuing SNMPv2C GetNextRequest to get ifIndex.10000.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	29	
	version	1(SNMPv2C)	02	01	01

	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	1C	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	11	
		variable-bindings		30	0F	
			name	1.3.6.1.2.1.2.2.1.1.10000 (ifIndex.10000)	06	0B	2b 06 01 02 01 02 02 01 01 CE 10
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	OC
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.2.2.1.2.[ifIndex]	06	*	2b 06 01 02 01 02 02 01 02 *
	value	Variable strings	04	*	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNextRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. value field is the correct ifDescr.[ifIndex] value and within the range field and syntax type

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.1.2.6 GetNext tabular objects with multiple OIDs

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNext tabular objects packet from multiple OIDs issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

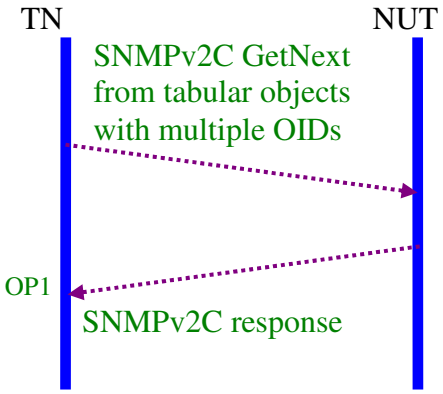
Refer Common Test Setup

Walk Preparation

Walk ifTable and udpTable

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest object to NUT by issuing SNMPv2C GetNextRequest to get tabular objects with multiple OIDs.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		Any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	37	
	version	1(SNMPv2C)	02	01	01

	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	2A	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	2C	
		variable-bindings		30	0B	
			name	1.3.6.1.2.1.2.2 (ifTable)	06	07	2b 06 01 02 01 01 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.2.2.1 (ifEntry)	06	08	2b 06 01 02 01 02 02 01
			value	NULL	05	00	
					30	0F	
			name	1.3.6.1.2.1.2.2.1.1.10000 (ifIndex.10000)	06	0B	2b 06 01 02 01 02 02 01 01 CE 10
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	01	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
		name	1.3.6.1.2.1.2.2.1.1.[ifIndex]	06	*	2b 06 01 02 01 02 02 01 01 *
		value	Integer32	02	*	*
				30	*	
		name	1.3.6.1.2.1.2.2.1.1.[ifIndex]	06	*	2b 06 01 02 01 02 02 01 01 *
		value	Integer32	02	*	*
				30	*	

			name	1.3.6.1.2.1.2.2.1.2.[ifIndex]	06	*	2b 06 01 02 01 02 02 01 02 *
			value	Variable strings of ifDescr	04	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNextRequest correctly.

Received packet with
1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the first returned variable binding's name is 1.3.6.1.2.1.2.2.1.1.ifIndex(IID=ifIndex) and its value is the IfIndex, the second returned variable binding is exactly the same with the first one and the third returned variable binding's name is 1.3.6.1.2.1.2.2.1.2.ifIndex(IID=ifIndex) and its value is the ifDescr

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.1.2.7 GetNext multiple OIDs from different tables

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetNextRequest object packet from different tables(the ifTable and udpTable) in the MIB issued by the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

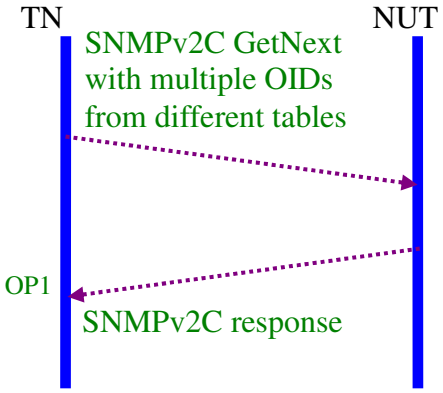
Refer Common Test Setup

Walk Preparation

Walk ifTable and udpTable

Procedure

The test sequence is as follows



1. TN send SNMPv2C GetNextRequest object to NUT by issuing SNMPv2C GetNextRequest to get ifType from ifTable and udpLocalAddress from udpTable.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	44	

	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	37	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	2C	
		variable-bindings		30	0D	
			name	1.3.6.1.2.1.2.2.1.3 (3:ifType)	06	09	2b 06 01 02 01 02 02 01 03
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	
					30	0D	
			name	1.3.6.1.2.1.7.5.1.1 *(1:udpLocalAddress)*	06	09	2b 06 01 02 01 07 05 01 01
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
		name	1.3.6.1.2.1.2.2.1.3.*[ifIndex]*	06	*	2b 06 01 02 01 02 02 01 03 *
		value	Integer32	02	*	*
				30	*	
		name	1.3.6.1.2.1.1.4.0	06	08	2b 06 01 02 01 01

				(sysContact.0)			04 00
			value	sysContact	04	*	*
					30	*	
			name	1.3.6.1.2.1.7.5.1.1.*[udpLocalAddress].[udpLocalPort]*	06	*	2b 06 01 02 01 07 05 01 01 *
			value	IPAddress	40	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetNextRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the first returned variable binding's name is first instance of ifType(1.3.6.1.2.1.2.2.1.3) entry, the second returned variable binding's name is 1.3.6.1.2.1.1.4.0 and its value is sysContact, the third returned variable binding's is first instance of udpLocalAddress(1.3.6.1.2.1.7.5.1.1) and its value is the udpLocalAddress.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.2 GetNext RequestID Correlation Check

Purpose

Verify that NUT playing the SNMPv2C agent can process each unique requestID from the GetNextRequest object packet issued by the SNMPv2C manager. Ten SNMP packets with continuous requested will be sent to NUT.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

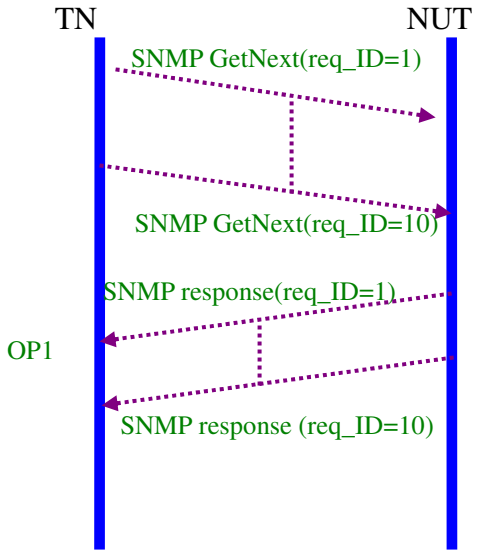
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest scalar object to NUT by issuing SNMPv2C GetNextRequest with requestID starting from 1 to 10.
2. NUT replies SNMPv2C Response with correct requestID to TN.

Sending packets

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	25	
	version			1(SNMPv2C)	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetNextRequest	A1	18	
		request-id		*(from 1 to 10)	02	01	*
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0D	
		variable-bindings			30	0B	
			name	1.3.6.1.2.1.1.3 (sysUpTime)	06	07	2b 06 01 02 01 01 03
			value	NULL	05	00	

Receiving packets

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			NUT_ADDRESS		
	Destination Address			TN_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in the sending corresponding packets		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	*	
		request-id	*(from 1 to 10)	02	01	*
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	0A	06 08 2b 06 01 02 01 01 03 00
		value	TimeTicks of sysUpTime	43	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address

TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received ten SNMPv2C responses from NUT responding to SNMPv2C GetNextRequest correctly.

The received 10 packets with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. The request-id should be 1 to 10 as the previously sent SNMPv2C GetNextRequest.
3. error-status must be equal to zero and error-index must be equal to zero

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2.

v6SNMPv2C2.3 Error Check

All error tests shall be followed by a SNMPv2C Get sysUpTime to check that the NUT is still functioning normally.

For detailed SNMP packet formats for Get sysUpTime, please refer to those listed in Pre-Test.

v6SNMPv2C2.3.1 GetNext with sequence_of error

v6SNMPv2C2.3.1.1 GetNext sequence_of type error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetNextRequest with invalid sequence_of type field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

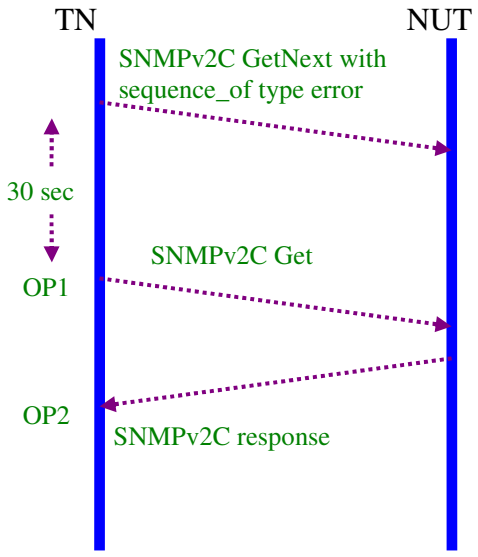
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows:



1. TN sends SNMPv2C GetNextRequest with sequence_of type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is

alive.

4. NUT returns correct sysUpTime

1st Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				00	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with sequence_of type error packet.

Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetNextRequest with sequence_of type error packet as should be expected.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.1.2 GetNext with sequence_of length Error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetNextRequest with invalid sequence_of length field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

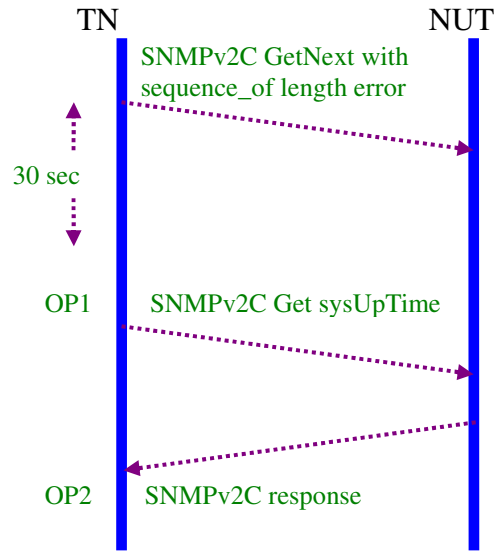
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest with sequence_of length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	***0F***	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with sequence_of length error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.2 GetNext with version number error
v6SNMPv2C2.3.2.1 GetNext with version number type error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetNextRequest with invalid version type field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

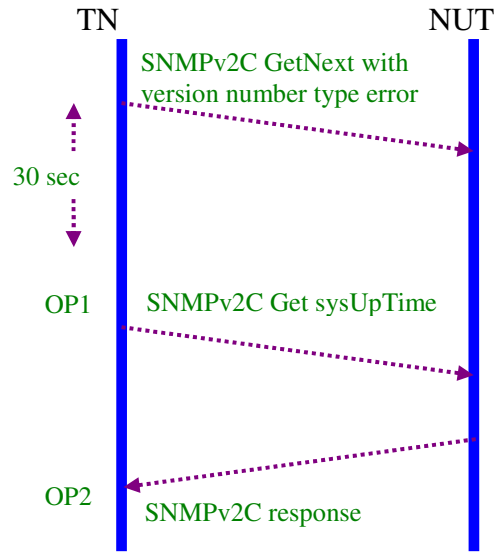
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest with version type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	***00***	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with version number type error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.2.2 GetNext version number length error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetNextRequest with invalid version length field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

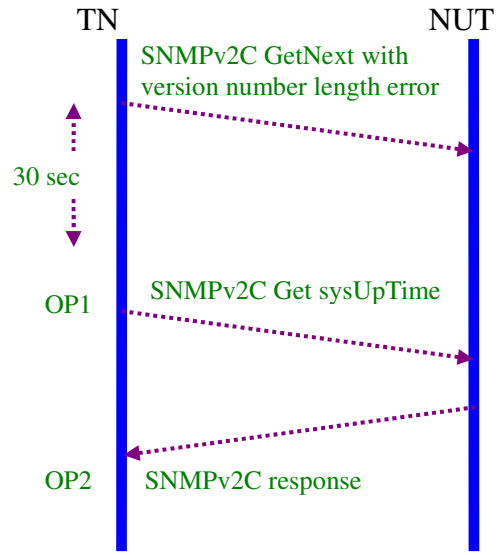
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with version number length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2C	02	***05***	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetNextRequest	A1	19	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with version number length error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.2.3 GetNext version number value error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetNextRequest with invalid version number value field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

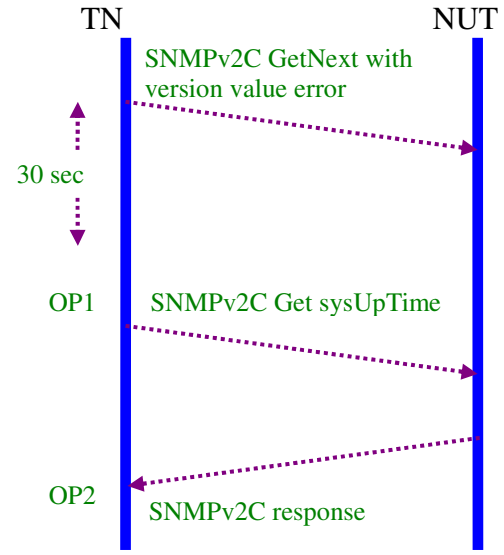
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with version number value error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	***20***
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and variable binding pair.

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with version value error packet.
OP2: TN received correct Response with SysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.3 GetNext with community error
v6SNMPv2C2.3.3.1 GetNext with community type error

Purpose

Verify that NUT playing as agent can properly detect the GetNextRequest with error community type in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

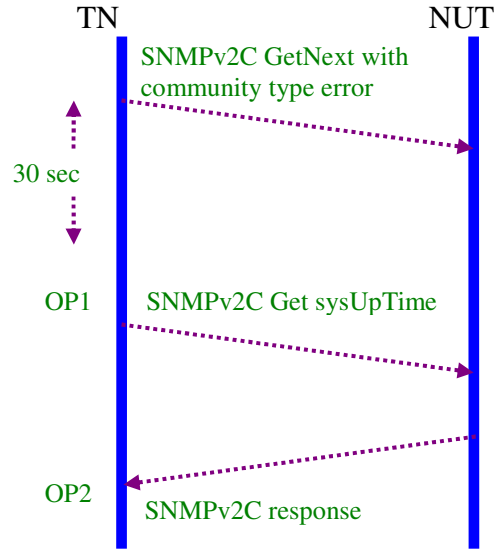
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with community type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	***02***	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with community type value error packet.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetNextRequest with community type value error packet as expected.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.3.2 GetNext with community length error

Purpose

Verify that NUT playing as agent can properly detect the GetNextRequest with error community length in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

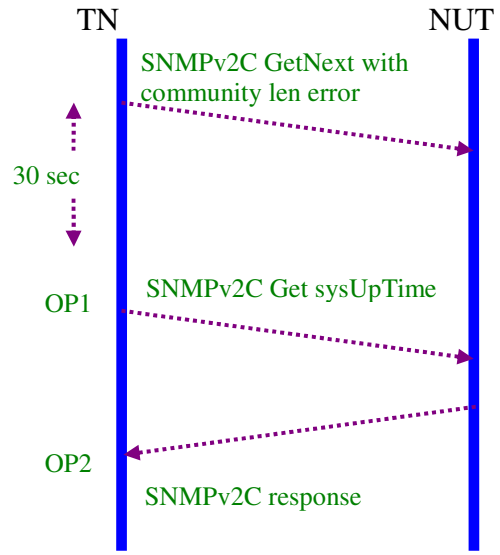
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with community length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2C	02	01	01
	community			public	04	***0F***	70 75 62 6C 69 63
	Data	PDU type		GetNextRequest	A1	19	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with community length error packet.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.3.3 GetNext with community value error

v6SNMPv2C2.3.3.3.1 Empty community_string

Purpose

Verify that NUT playing as agent can properly detect the GetNextRequest with empty community string in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

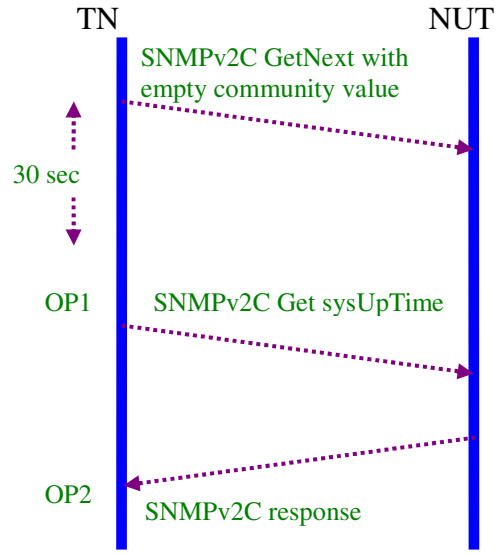
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with empty community string to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			Any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community			***04***	***06***	
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Judgment

OP1: NUT will silently discard this empty community string SNMPv2C GetNextRequest

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.3.3.2 Inconsistent community_string

Purpose

Verify that NUT playing as agent can properly detect the GetNextRequest with inconsistent community string in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

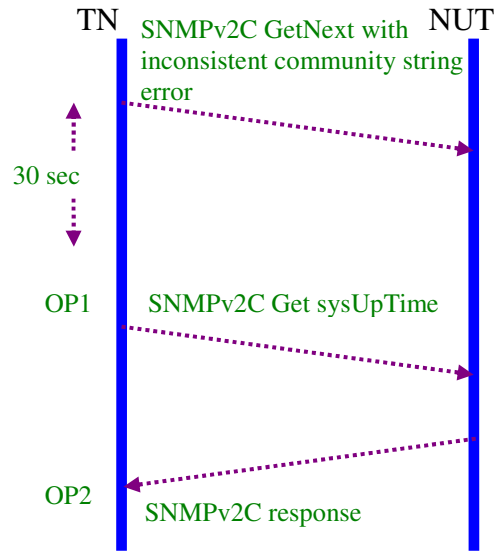
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with inconsistent community string to NUT.
2. NUT discards the datagram and continues to respond to normal requests
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		1(SNMPv2C)	02	01	01
	community		*pupuic* (public)	04	06	70 75 ***70 75*** 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Judgment

OP1: NUT will silently discard this SNMPv2C GetNextRequest with empty community string value error packet.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.3.3.3 community_string_with_CarriageReturn_LineFeed

Purpose

Verify that NUT playing as agent can properly detect the GetNextRequest with carriage return and linefeed error community value in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

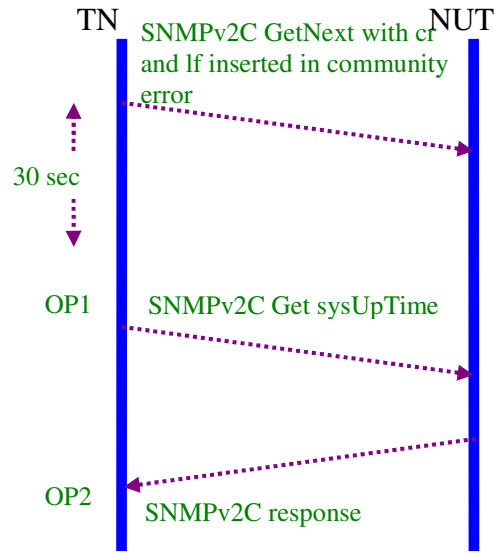
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with carriage return and line feed inserted in the community string error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		Public	04	06	70 75 ***0d 0a*** 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Judgment

OP1: NUT will silently discard this SNMPv2C GetNextRequest with CRLF inserted in community string value error packet.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.4. GetNext with PDU error

v6SNMPv2C2.3.4.1 GetNext with PDU length error

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetNextRequest with invalid PDU length field in the received packet from the SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

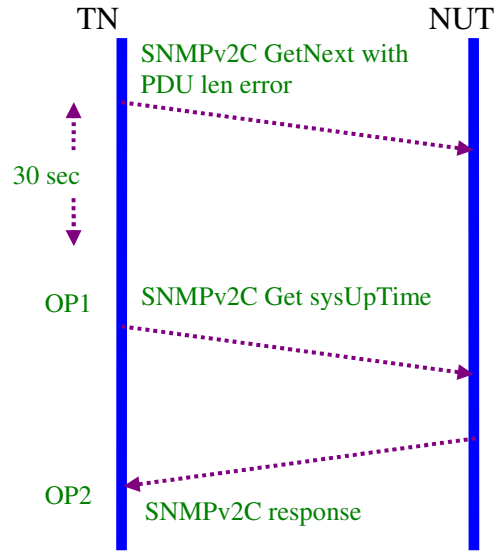
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetNextRequest with PDU length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	***00***	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid PDU length error packet.
OP2 TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.5 GetNext with request ID error

v6SNMPv2C2.3.5.1 GetNext with request ID type error

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with request ID field type error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

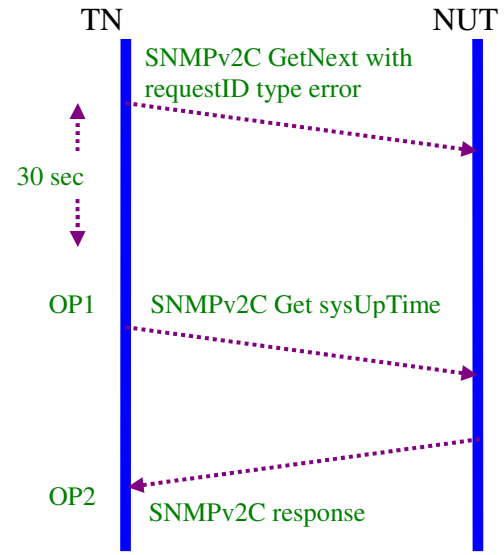
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with request ID type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	***20***	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with request ID type error packet.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetNextRequest with request ID type error packet as expected.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.5.2 GetNext with request ID length error

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with request ID field length error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

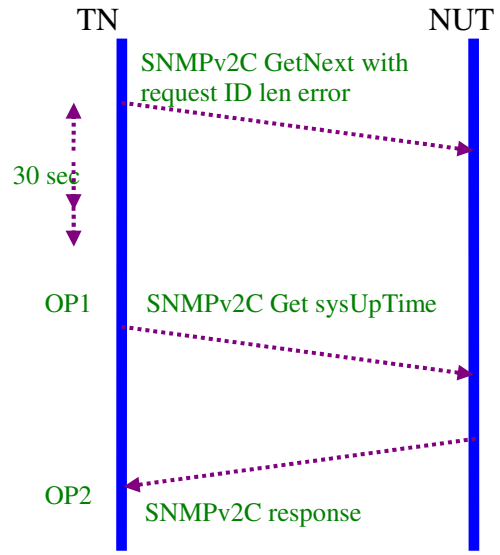
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with request ID length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2C	02	01	01
	community			pubic	04	06	70 75 62 6C 69 63
	Data	PDU type		GetNextRequest	A1	19	
		request-id		12	02	***0F***	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid request ID len error packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.5.3 GetNext with request ID value error
v6SNMPv2C2.3.5.3.1 GetNext with requestID greater than maximum value(214783647,0x0CCD569F)

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with requestID value exceeding the maximum possible value in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

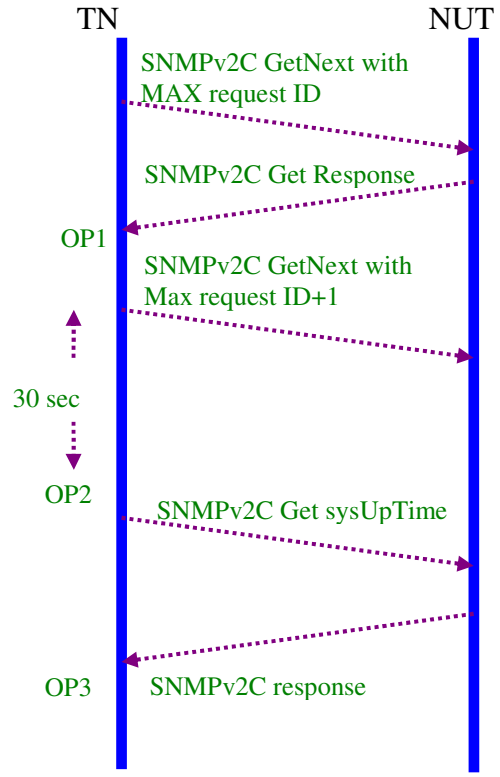
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with request ID value =214783647 to NUT.
2. NUT responds with Response.
3. TN sends SNMPv2C GetNextRequest with request ID value =214783648 error to NUT.
4. NUT discards the datagram and continues to respond to normal requests.
5. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is

alive

6. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	214783647	02	***04***	***0C CD 56 9F***
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 00
			value	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		Public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	*	
		request-id	214783647	02	***04***	***0C CD 56 9F***
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		var		30	*	

		iab le-bin din gs	name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value	Octet string of NUT system contact	04	*	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)							
IP Header	Source Address			TN_ADDRESS			
	Destination Address			NUT_ADDRESS			
UDP Header	Source Port			any			
	Destination Port			161			
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)			
				type	len	value	
				30	29		
	version		SNMPv2C	02	01	01	
	community		public	04	06	70 75 62 6C 69 63	
	Data	PDU type	GetNextRequest	A1	1C		
		request-id	214783648	02	***04***	***0C CD 56 A0***	
		error-status	0	02	01	00	
		error-index	0	02	01	00	
				30	0E		
		var iab le-bin din gs		30	0C		
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT returns normal Response with correct sysUpTime
OP2: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid request ID error packet.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetNextRequest with invalid request ID error packet as expected.
OP3: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.1

v6SNMPv2C2.3.5.3.2 GetNext with requestID smaller than the minimum value(-214783648, F332A960)

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with request ID value= -214783648 error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

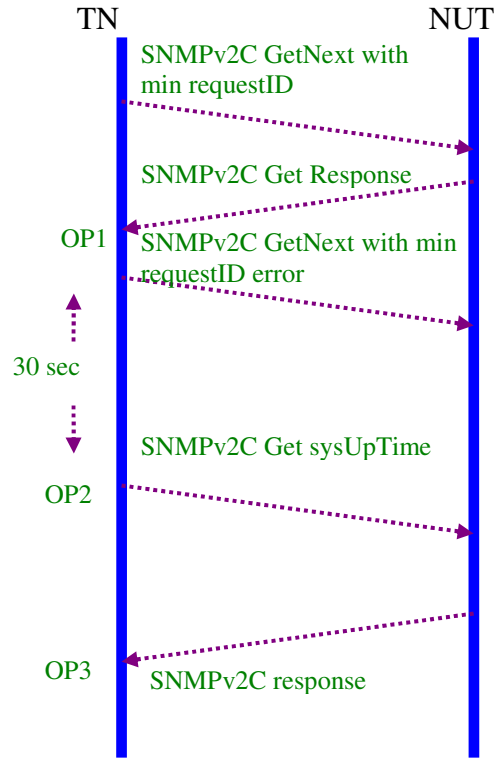
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with request ID value =-214783648 to NUT.
2. NUT responds with Response.
3. TN sends SNMPv2C GetNextRequest with request ID value =-214783649 error to NUT.
4. NUT discards the datagram and continues to respond to normal requests.

5. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
6. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	-214783648	02	***04***	***F3 32 A9 60***
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		Public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	*	
		request-id	-214783648	02	***04***	***F3 32 A9 60***
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	

		variable-bindings			30	*	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	TimeTicks of NUT system up time	43	*	TimeTicks

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	29	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	1C	
		request-id	-214783649	02	***04***	***F3 32 A9 5F***
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT returns normal Response with correct sysUpTime
OP2: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid request ID packet.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetNextRequest with invalid request ID error packet as expected.
OP3: TN receives correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management

Protocol, Sec 4.1

v6SNMPv2C2.3.6 GetNext with error-status error
v6SNMPv2C2.3.6.1 GetNext with error-status type error

Purpose

Verify that NUT playing as agent can properly detect the GetNextRequest with error-status type BER error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

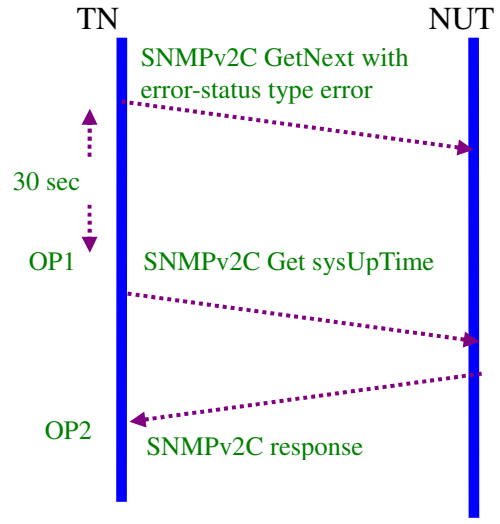
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with error-status type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	26	
	version			SNMPv2C	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetNextRequest	A1	19	
		request-id		12	02	01	0C
		error-status		0	***20***	01	00
		error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid error status type packet.
Note : Warning will be the test judgement when NUT does not discard this malformed GetNextRequest with invalid error status type packet as expected.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 3

v6SNMPv2C2.3.6.2 GetNext with error-status length error

Purpose

Verify that NUT playing as agent can properly detect the GetNextRequest with error-status length error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

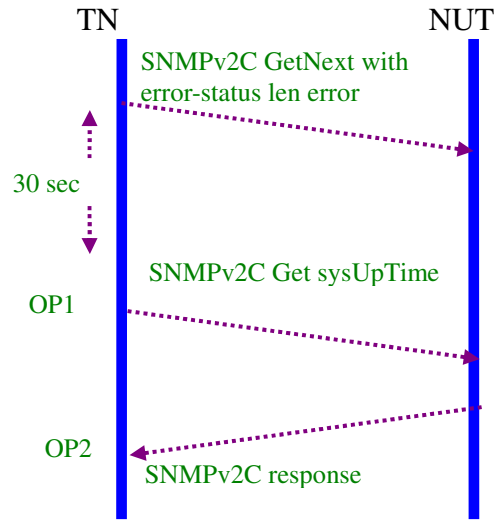
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with error-status length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	***10***	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid error status length packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.6.3 GetNext with error-status non-zero error

Purpose

Verify that NUT playing as agent can properly detect the GetNextRequest with error-status value none zero error in the received packet from SNMPv2C manager and will ignore this eror-status non-zero datagram and respond with correct OID next to sysUpTime.0.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

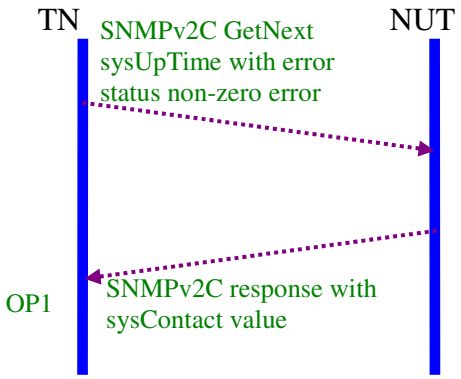
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with non-zero error-status error to NUT.
2. NUT returns correct sysContact value.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		Public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	16	02	01	***10***

	a	error-index		0	02	01	00
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			NUT_ADDRESS		
	Destination Address			TN_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value	sysContact value	4	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received correct Response with sysContact value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2 PDU processiong.End of the Paragraph 1 “For example, some PDUs

(e.g., the GetRequest-PDU) are concerned only with the name of a variable and not its value".

v6SNMPv2C2.3.7 GetNext with error-index error
v6SNMPv2C2.3.7.1 GetNext with error-index type error

Purpose

Verify that NUT playing as agent can properly detect the SNMPv2C GetNextRequest with error-index type error. in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

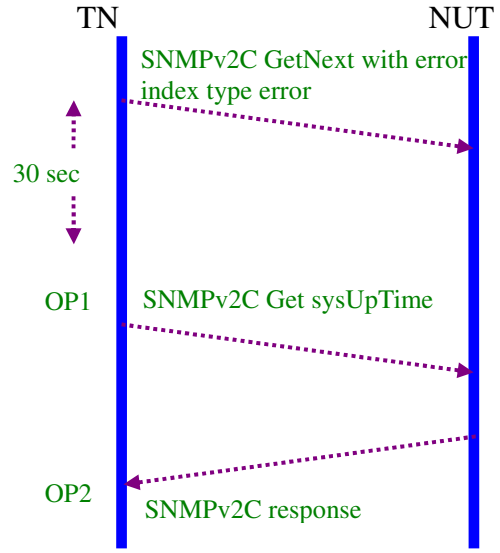
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with error-index type error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	***20***	01	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid error-index type packet.
Note : Warning will be the test judgement when NUT does not discard this malformed SNMPv2C GetNextRequest with invalid error-index type packet as expected.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.7.2 GetNext with error-index length error

Purpose

Verify that NUT playing as SNMPv2C agent can properly detect SNMPv2C GetNextRequest with error-index length error. In the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

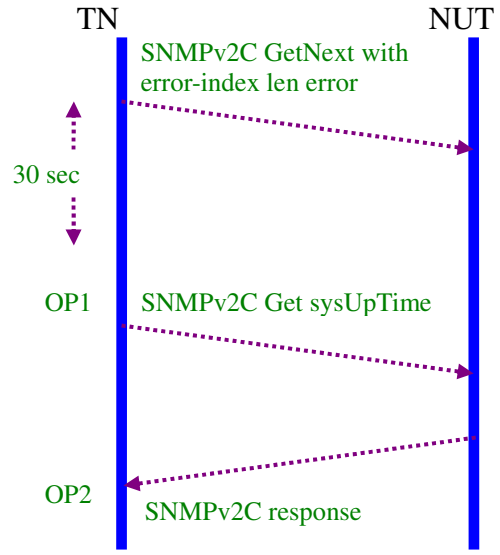
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with error-index length error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		pubic	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	***10***	00
				30	0E	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid error-index length error packet.
OP2: TN received correct Response with the sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.7.3 GetNext with error-index non-zero error

Purpose

Verify that NUT playing as SNMPv2Cagent can properly detect SNMPv2C GetNextRequest with error-index non-zero error in the received packet from SNMPv2C manager and will ignore this error packet and return correct packet.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

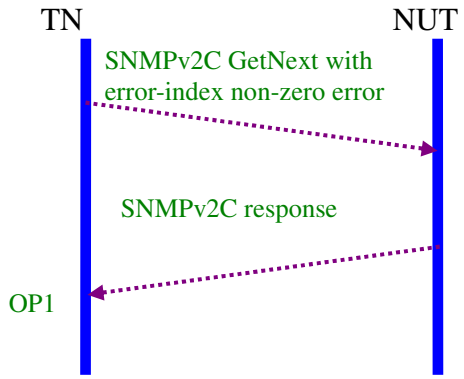
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with error-index non-zero error to NUT.
2. NUT returns correct sysContact value.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address		TN_ADDRESS			
	Destination Address		NUT_ADDRESS			
UDP Header	Source Port		any			
	Destination Port		161			
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)			
			type	len	value	
			30	26		
	version	SNMPv2C	02	01	01	
	community	public	04	06	70 75 62 6C 69 63	
	Da PDU type	GetNextRequest	A1	19		
	Da request-id	12	02	01	0C	

	t a	error-status		0	02	01	00
		error-index		16	02	01	***10***
					30	0E	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime)	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
	value	sysContact value	4	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received correct Response with the sysContact value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management

Protocol, Sec 4.2

v6SNMPv2C2.3.8 GetNext with variable-bindings error
v6SNMPv2C2.3.8.1 GetNext with OID type error

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with OID encoding type error of variable binding's name in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

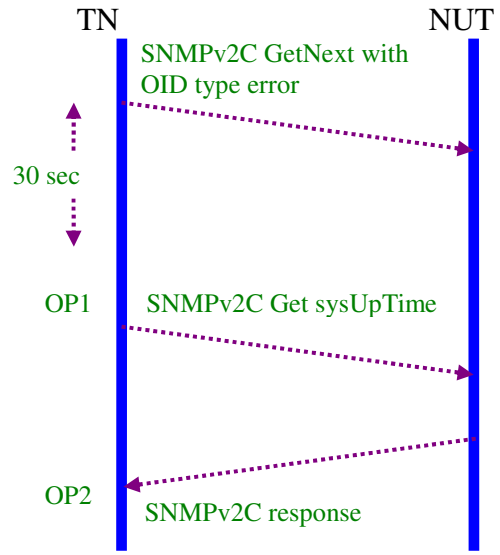
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with variable binding type value error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	***07***	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid OID type packet.
OP2: TN received correct Response with the current sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.8.2 GetNext with OID length error

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with OID len error in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

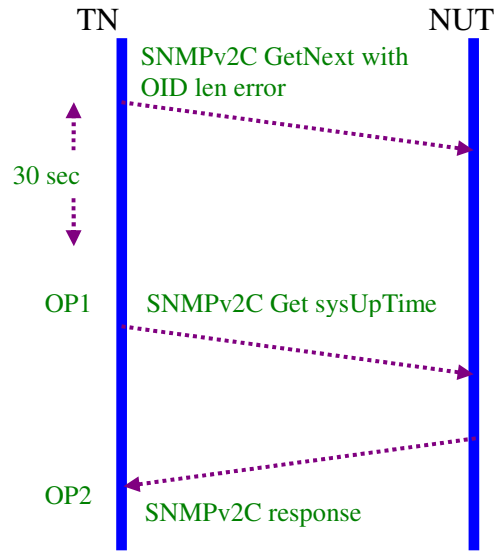
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with OID type value error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	26	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	***18***	2b 06 01 02 01 01 03 00
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with invalid OID length packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.8.3 GetNext with OID value error

v6SNMPv2C2.3.8.3.1 GetNext with FF value in variable-binding's name

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with FF OID value error in the received packet from SNMPv2C manager and will respond tooBig error code.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

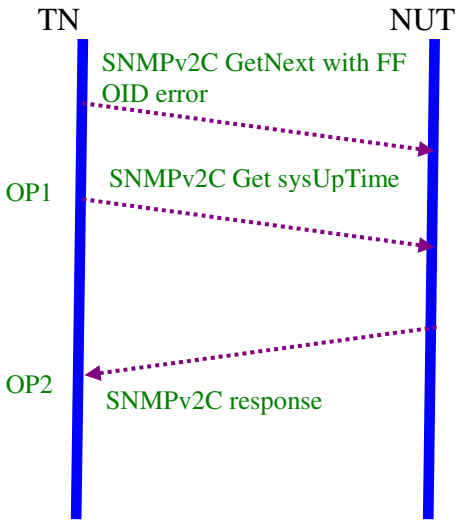
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with FF value in OID.
2. NUT silently discards this error packet.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	*	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	01	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	06	08	2b 06 01 02 01 01 03 ***FF***
			value	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will respond with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status must be equal to one(tooBig) and error-index must be equal zero
4. empty variable binding's field.

OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

v6SNMPv2C2.3.8.3.2 GetNext with variable-binding's value without NULL

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with variable binding's value without NULL in the received packet from SNMPv2C manager and will discard the datagram and continue to respond to normal requests.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

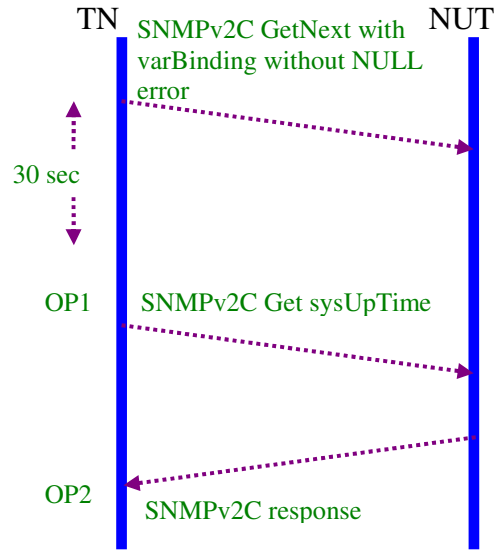
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with variable binding's value without NULL value error to NUT.
2. NUT discards the datagram and continues to respond to normal requests.
3. TN sends SNMPv2C Get sysUpTime to NUT for checking the SNMP agent is alive.
4. NUT returns correct sysUpTime.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	any

Header	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	24	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	17	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0C	
		variable-bindings		30	0A	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetNextRequest with variable binding's value without NULL packet.
OP2: TN received correct Response with sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol

v6SNMPv2C2.3.8.3.3 GetNext with zero variable-bindings

Purpose

Verify that SNMPv2C agent can properly detect the GetNextRequest with zero variable-bindings in the received packet from SNMPv2C manager and will reply with Response with empty variable.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

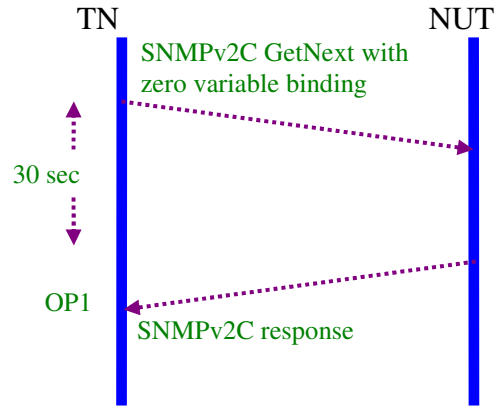
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with zero variable binding to NUT.
2. NUT will replies with Response with empty variable binding.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63

	D a t a	PDU type		GetNextRequest	A1	0B	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	00	
		variable-bindings					
			name				
			value				

2nd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data: PDU type	Response	A2	0B	
	Data: request-id	12	02	01	0C
	Data: error-status	0	02	01	00
	Data: error-index	0	02	01	00
	Data:		30	00	
	Data: variable-bindings				
	Data: variable-bindings: name				
	Data: variable-bindings: value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will reply with SNMPv2C correct Response with zero variable binding.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

v6SNMPv2C2.3.8.3.4 128 sub_identifiers check

Purpose

Verify that SNMPv2C agent can properly handle GetNextRequest with 128 sub-identifiers in the received packet from SNMPv2C manager and will respond endOfMIB.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

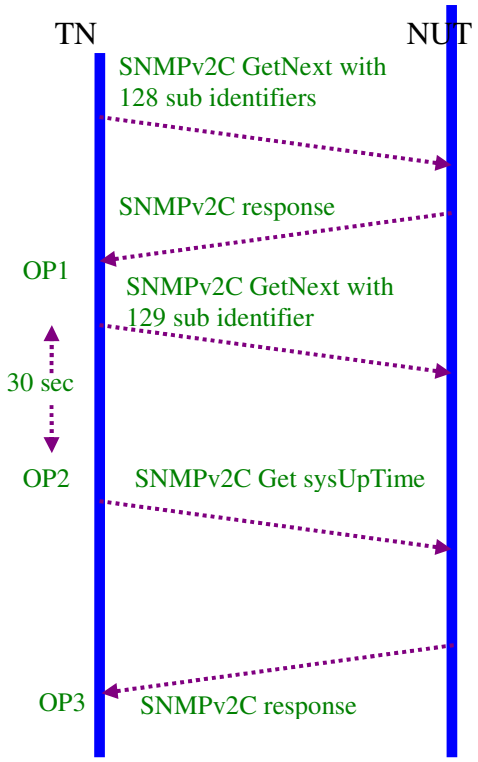
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with 128 sub identifiers error to NUT.
2. NUT responses with endOfMIBView
3. TN sends SNMPv2C GetNextRequest with 129 sub identifiers to NUT.
4. NUT discards the datagram and continues to respond to normal requests.
5. TN sends SNMPv2C Get sysUpTime to NUT for verifying the NUT is still alive
6. NUT returns current sysUpTime value

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)							
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	81 A0	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	81 92	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	81 86	
		variable-bindings		30	81 83	
			name	2.3.6.1.2.3.4.5.6.7.8…..125 (128 sub IDs)	06	***127***	53 06 01 02 01 03 04 05 06 07 …..7D
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	81 A0	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A1	81 92	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	81 86	

		variable-bindings			30	81 83	
			name	2.3.6.1.2.3.4.5.6.7.8…..125 (128 sub IDs)	06	***127***	53 06 01 02 03 04 05 06 07 …..7D
			value	***endofMIBView***	***82***	***00***	

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	81 A2	
	version		SNMPv2C	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetNextRequest	A1	81 94	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	81 88	
		variable-bindings		30	81 85	
			name	2.3.6.1.2.3.4.5.6.7.8…..126 (129 sub IDs)	06	81 80	53 06 01 02 03 04 05 06 07 …..7E
			value	NULL	05	00	

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will respond with EndOfMIB Response.
OP2: NUT silently drops this 129 sub identifier error packet
OP3: NUT receives the current sysUpTime value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management

Protocol, Sec. 4.1
RFC 2578, Structure of Management Information Version 2 (SMIv2), Sec 3.5

v6SNMPv2C2.4 GetNext with tooBig message

Purpose

Verify that SNMPv2C agent can properly handle the large GetNextRequest in the received packet from SNMPv2C manager and will respond either Response normally or with tooBig error code when the size of the resultant message is less than or equal to both a local constraint and the maximize size of the SNMPv2C manager.

.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

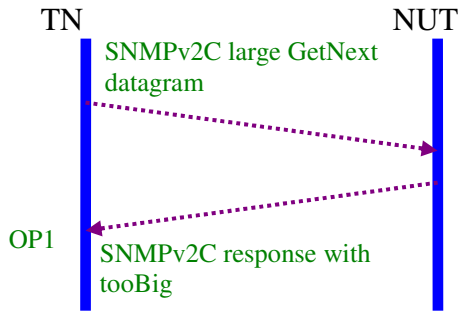
Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure



1. TN sends SNMPv2C GetNextRequest with variable binding exceeding the maximum value to NUT.
2. NUT responds with either with normal Response or with tooBig error code to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	820594(1428)	
	version	SNMPv2C	02	01	01
	community	public	04	06	70 75 62 6C 69 63

	D a t a	PDU type		GetNextRequest	A1	8205 85(1 413)	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	***82 0578***	***05 DC(for 100 variable-bindings)***
		var iab le- bin din gs			30	0C	
			name	1.3.6.1.2.1.1.3.0	06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	
					30	0C	
			name		06	08	2b 06 01 02 01 01 03 00
			value	NULL	05	00	
			name				Until 100 repetitions
			value				

2nd Packet is either normal Response

Standard query from SNMP manager (NUT) to SNMP agent (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in the 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	D a t a PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	1	02	01	01
	error-index	0	02	01	00
			30	*	
	name	1.3.6.1.2.1.1.3.0	06	08	2b 06 01 02 01 01 03 00
	value	TimeTicks	*	*	
			30	*	
	name	1.3.6.1.2.1.1.3.0	06	08	2b 06 01 02 01 01 03 00
	value	TimeTicks	*	*	Repeat until 100 repetitions
	name				

			value				

Or Response with error-status=tooBig

Standard query from SNMP manager (NUT) to SNMP agent (TN)							
IP Header	Source Address			NUT_ADDRESS			
	Destination Address			TN_ADDRESS			
UDP Header	Source Port			161			
	Destination Port			Same as the source port in the 1st packet			
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)			
				type	len	value	
				30	18		
	version		1(SNMPv2C)	02	01	01	
	community		public	04	06	70 75 62 6C 69 63	
	Data	PDU type	Response	A2	0B		
		request-id	12	02	01	0C	
		error-status	1	02	01	01(tooBig)	
		error-index	0	02	01	00	
				30	00		
		name					
		value					

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: NUT will respond with
1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetNextRequest
3. error-status and error-index must be equal zero
4. the variable binding is the 100 variable binding listed in the first GetNextRequest.

Or OP1: NUT will respond with
1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type = A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to one(tooBig) and error-index must be equal zero
4. empty variable binding's field.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.2

Group 3 IPv6 SNMPv2C GetBulkRequest

Scope

The following tests verify the GetBulkRequest commands in the SNMPv2C protocol.

Overview

Tests in this group verify that a SNMPv2C agent node can properly process and generate the correct SNMPv2C messages with Response PDUs according to the SNMPv2C GetBulk commands from the SNMPv2C manager using the N+(M*R) relationship where N is the minimum of: a) the value of the non-repeaters field in the request, and b) the number of variable-bindings in the request; M is the value of the max-repetitions field in the request; and R is the maximum of: a) number of variable-bindings in the request - N, and b) zero as defined in GetBulk protocol operations. These tests also verify a SNMPv2C agent node will transmit the appropriate SNMPv2C parameter problem error messages in response to invalid or unknown fields in the received SNMPv2C packets.

v6SNMPv2C3.1 GetBulk with zero non-repeaters, zero max-repetitions and zero variable-bindings

purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with zero noon-repeaters, zero max-repetitions and zero variable binding packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

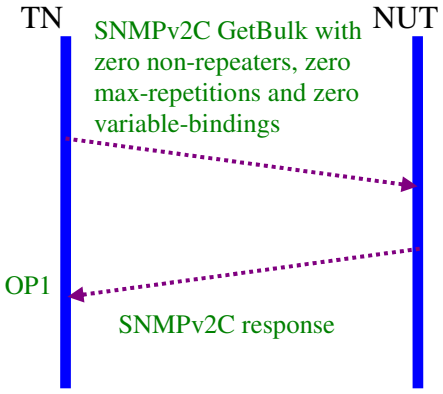
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with zero non-repeaters, zero max-repetitions and zero variable-bindings.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	Version	SNMPv2C	02	01	01

	Community			public	04	06	70 75 62 6C 69 63
	D a t a	PDU type		GetBulkRequest	A5	0B	
		request-id		12	02	01	0C
		non-repeaters		0	02	01	00
		max-repetitions		0	02	01	00
					30	00	
		variable-bindings					
			name				
			value				

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	D a t a PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
	name				
	value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received Packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetBulkRequest
3. error-status must be equal to zero and error-index must be equal zero
4. The variable binding list should be empty.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol Sec.4.2.3

v6SNMPv2C3.2 GetBulk with zero non-repeaters, non-zero max-repetitions and zero variable-bindings

purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with zero non-repeaters, non-zero max-repetitions and zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

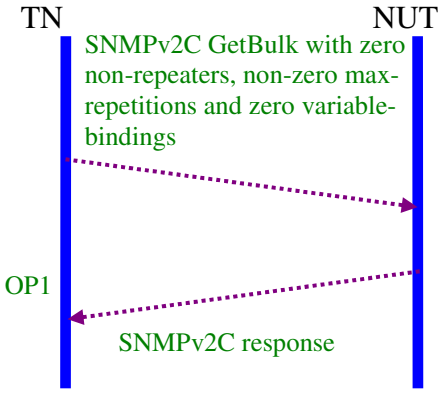
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with zero for non-repeaters and variable-bindings , non-zero(2 in this test) for max-repetitions.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	Version	SNMPv2C	02	01	01

	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	0B	
		request-id	12	02	01	0C
		non-repeaters	0	02	01	00
		max-repetitions	2	02	01	02
				30	00	
		variable-bindings				
			name			
			value			

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
		name			
		value			

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN Received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received Packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetBulkRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.3 GetBulk with non-zero non-repeaters, zero max-repetitions and zero variable-bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with non-zero noon-repeaters, zero max-repetitions and zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

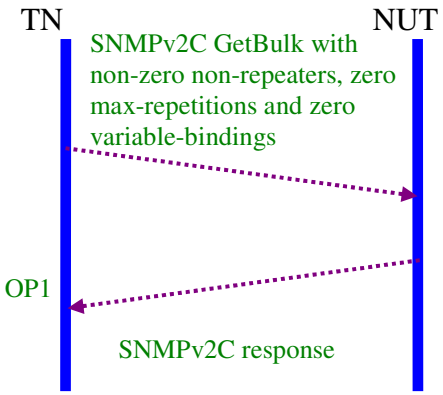
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with non-zero value (2 in this test) for non-repeaters and zero for max-repetitions and variable binding.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Messag	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

e				30	18	
	Version		SNMPv2C	02	01	01
	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	0B	
		request-id	12	02	01	0C
		non-repeaters	2	02	01	02
		max-repetitions	0	02	01	00
				30	00	
		variable-bindings				
			name			
			value			

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
		name			
		value			

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.4 GetBulk with non-zero non-repeaters, non-zero max-repetitions and zero variable-bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with non-zero non-repeaters, non-zero max-repetitions and zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

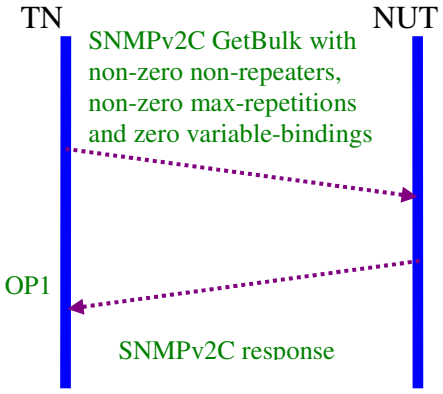
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with non-zero value for non-repeaters and max-repetitions (2 in this test) and zero for variable-bindings.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	

	Version		SNMPv2C	02	01	01
	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	0B	
		request-id	12	02	01	0C
		non-repeaters	2	02	01	02
		max-repetitions	2	02	01	02
				30	00	
		variable-bindings				
		name				
		value				

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
	name				
	value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C

GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.5 GetBulk with zero non-repeaters, zero max-repetitions and non-zero variable-bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with zero non-repeaters, zero max-repetitions and non-zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

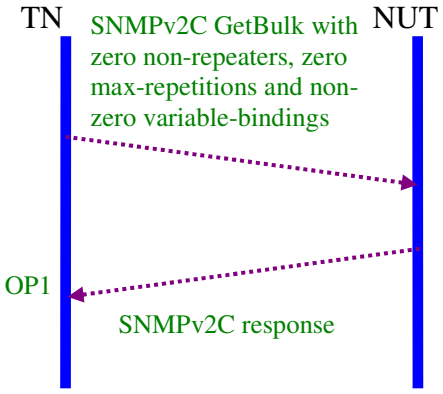
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with zero for non-repeaters and max-repetitions (1 in this test) and 7 for variable-bindings in system group.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	73	

Version			SNMPv2C	02	01	01
Community			public	04	06	70 75 62 6C 69 63
Data	PDU type		GetBulkRequest	A5	66	
	request-id		12	02	01	0C
	non-repeaters		0	02	01	00
	max-repetitions		0	02	01	00
				30	5B	
	variable-bindings			30	0B	
		name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.2 (sysObjectID)	06	07	2b 06 01 02 01 01 02 00
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.3 (sysUpTime)	06	07	2b 06 01 02 01 01 03 00
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.4 (sysContact)	06	07	2b 06 01 02 01 01 04
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.5 (sysName)	06	07	2b 06 01 02 01 01 05
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.6 (sysLocation)	06	07	2b 06 01 02 01 01 06
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.7 (sysServices)	06	07	2b 06 01 02 01 01 07
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

				30	18	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	0B	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	00	
		variable-bindings				
		name				
		value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulkRequest request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.6 GetBulk with non-zero non-repeaters, zero max-repetitions and non-zero variable-bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with non-zero non-repeaters, zero max-repetitions and non-zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

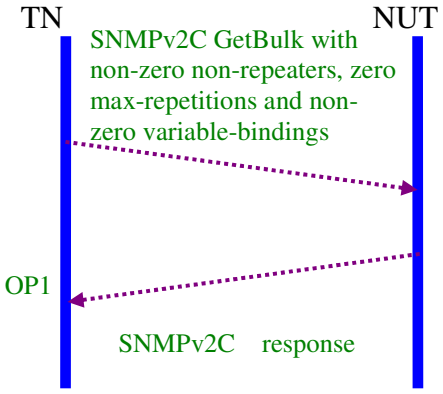
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest Scalar object to NUT by issuing SNMPv2C GetBulkRequest with non-zero values for non-repeaters (2 in this test) and zero for max-repetitions and 7 for variable-bindings in system group.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	73	

Version			SNMPv2C	02	01	01
Community			public	04	06	70 75 62 6C 69 63
Data	PDU type		GetBulkRequest	A5	67	
	request-id		12	02	01	0C
	non-repeaters		2	02	01	02
	max-repetitions		0	02	01	00
				30	5B	
	variable-bindings			30	0B	
		name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.2 (sysObjectID)	06	07	2b 06 01 02 01 01 02
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.3 (sysUpTime)	06	07	2b 06 01 02 01 01 03
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.4 (sysContact)	06	07	2b 06 01 02 01 01 04
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.5 (sysName)	06	07	2b 06 01 02 01 01 05
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.6 (sysLocation)	06	07	2b 06 01 02 01 01 06
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.7 (sysServices)	06	07	2b 06 01 02 01 01 07
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the 1st packet source port		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

				30	*	
version			1(SNMPv2C)	02	01	01
community			public	04	06	70 75 62 6C 69 63
Data	PDU type		Response	A2	*	
	request-id		12	02	01	0C
	error-status		0	02	01	00
	error-index		0	02	01	00
				30	*	
	variable-bindings			30	*	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	octet string of NUT system description	04	*	variable string*
				30	*	
		name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
		value	Object identifier of NUT system objectID	06	*	variable object identifier*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the first name/value pair is 1.3.6.1.2.1.1.1.0(sysDescr.0) and 1.3.6.1.2.1.1.2.0(sysObjectID.0) respectively and their values should be with correct syntax types and within their defined value ranges

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol Sec.4.2.3

v6SNMPv2C3.7 GetBulk with zero non-repeaters, non-zero max-repetitions and non-zero variable-bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with zero non-repeaters, non-zero max-repetitions and non-zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

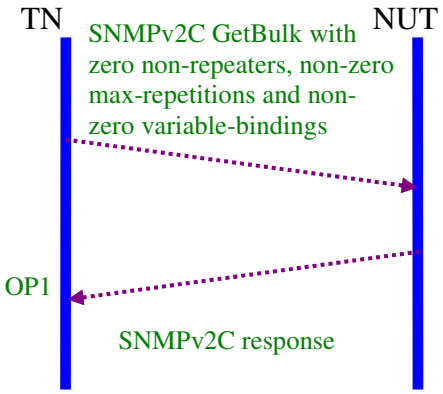
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with zero value for non-repeaters, non zero for max-repetitions (1 in this test) and 7 for variable binding in system group.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	73	

Version			1(SNMPv2C)	02	01	01
Community			public	04	06	70 75 62 6C 69 63
Data	PDU type		GetBulkRequest	A5	66	
	request-id		12	02	01	0C
	non-repeaters		0	02	01	00
	max-repetitions		1	02	01	01
				30	5B	
	variable-bindings			30	0B	
		name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.2 (sysObjectID)	06	07	2b 06 01 02 01 01 02
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.3 (sysUpTime)	06	07	2b 06 01 02 01 01 03
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.4 (sysContact)	06	07	2b 06 01 02 01 01 04
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.5 (sysName)	06	07	06 08 2b 06 01 02 01
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.6 (sysLocation)	06	07	2b 06 01 02 01 01 06
		value	NULL	05	00	
				30	0B	
		name	1.3.6.1.2.1.1.7 (sysServices)	06	07	2b 06 01 02 01 01 07
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

				30	*	
version			1(SNMPv2C)	02	01	01
community			public	04	06	70 75 62 6C 69 63
Data	PDU type		Response	A2	*	
	request-id		12	02	01	0C
	error-status		0	02	01	00
	error-index		0	02	01	00
				30	*	
	variable-bindings			30	*	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	octet string of NUT system description	04	*	variable string*
				30	*	
		name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
		value	Object identifier of NUT system objectID	06	*	variable object identifier*
				30	*	
		name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
		value	Time ticks of NUT system up time	43	*	variable time ticks*
				30	*	
			1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			octet string of NUT system contact information	04	*	variable string*
		name		30	*	
		value	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01
			octet string of NUT system name	04	*	variable string*
				30	0C	
		name	1.3.6.1.2.1.1.6.0 (sysLocation.0)	06	08	2b 06 01 02 01 01 06 00
		value	NUT's sysLocation	04	*	*
				30	0C	
		name	1.3.6.1.2.1.1.7.0 (sysServices.0)	06	08	2b 06 01 02 01 01 07 00

			value	NUT's sysServices	02	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulkRequest request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned variable-bindings list is 1.3.6.1.2.1.1.1.0(sysDescr.0),1.3.6.1.2.1.1.2.0(sysObjectID.0),1.3.6.1.2.1.1.3.0(sysUpTime.0),1.3.6.1.2.1.1.4.0(sysContact.0),1.3.6.1.2.1.1.5.0(sysName.0),1.3.6.1.2.1.1.6.0(sysLocation.0),1.3.6.1.2.1.1.7.0(sysServices.0) respectively and their values should be with correct syntax types and within their defined value ranges.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.8 GetBulk with non-zero non-repeaters, non-zero max-repetitions and non-zero variable bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with non-zero values for non-repeaters, max-repetitions and variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

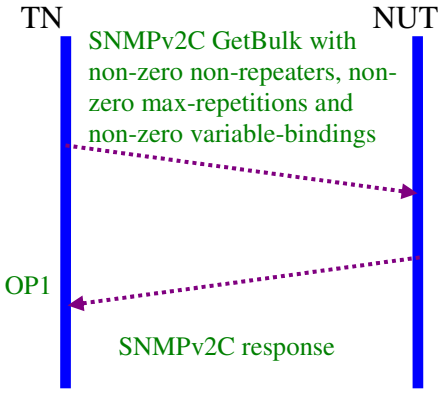
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with non-zero value for non-repeaters, max-repetitions (2 in this test) and SNMP system group OIDs(including 1.3.6.1.2.1.1.1(sysDescr), 1.3.6.1.2.1.1.2(sysObjectID) 1.3.6.1.2.1.1.3(sysUpTime) 1.3.6.1.2.1.1.4(sysContact)) for variable-bindings in this packet.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	4C	
	Version		1(SNMPv2C)	02	01	01
	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	34	
		request-id	12	02	01	0C
		non-repeaters	2	02	01	02
		max-repetitions	3	02	01	03
				30	34	
		variable-bindings		30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.2 (sysObjectID)	06	07	2b 06 01 02 01 01 02
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.3 (sysUpTime)	06	07	2b 06 01 02 01 01 03
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.4 (sysContact)	06	07	2b 06 01 02 01 01 04
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	*	
	var		30	*	

		iab le-bin dings	name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	octet string of NUT system description	04	*	variable string*
					30	*	
			name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
			value	Object identifier of NUT system objectID	06	*	variable object identifier*
					30	*	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	timeTicks	43	*	*
					30	*	
			name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value	octet string of NUT system contact information	04	*	variable string*
					30	*	
			name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value	octet string of NUT system contact information	04	*	variable string*
					30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	NUT system name	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	NUT system name	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.6.0 (sysLocation.0)	06	08	2b 06 01 02 01 01 06 00
			value	NUT system location	02	01	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned variable-bindings list is 1.3.6.1.2.1.1.1.0(sysDescr.0), 1.3.6.1.2.1.1.2.0(sysObjectID.0), 1.3.6.1.2.1.1.3.0(sysUpTime.0), 1.3.6.1.2.1.1.4.0(sysContact.0), 1.3.6.1.2.1.1.4.0(sysContact.0), 1.3.6.1.2.1.1.5.0(sysName.0), 1.3.6.1.2.1.1.5.0(sysName.0), 1.1.3.6.1.2.1.1.6.0(sysLocation.0) respectively and their values should be with correct syntax types and within their defined value ranges.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.9 GetBulk with negative non-repeaters, zero max-repetitions and zero variable bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with negative values for non-repeaters, zero values for max-repetitions and variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

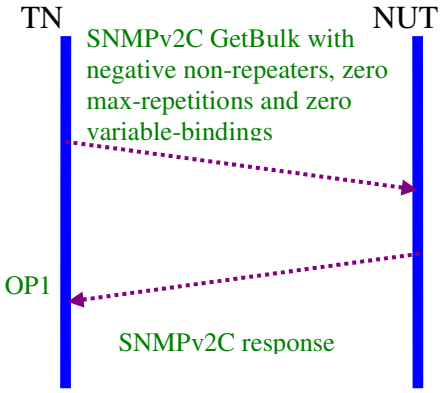
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with -1 for non-repeaters and zero for max-repetitions and variable-bindings.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	

	Version		1(SNMPv2C)	02	01	01
	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	0B	
		request-id	12	02	01	0C
		non-repeaters	-1	02	01	***FF***
		max-repetitions	0	02	01	00
				30	00	
		variable-bindings				
			name			
			value			

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
		name			
		value			

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C

GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.10 GetBulk with zero non-repeaters, negative max-repetitions and zero variable bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with zero for non-repeaters and variable-bindings, negative values for max-repetitions packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

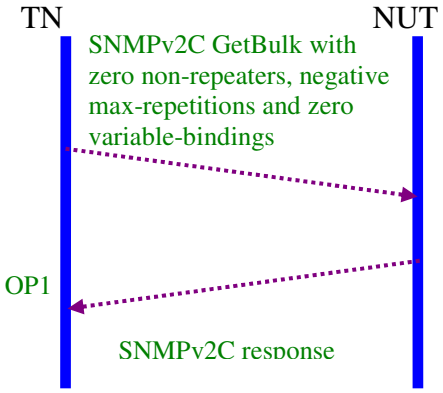
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest object to NUT by issuing SNMPv2C GetBulkRequest with -1 for max-repetitions and 0 for non-repeater and variable-binding.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Messag	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

e				30	18	
	Version		1(SNMPv2C)	02	01	01
	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	0B	
		request-id	12	02	01	0C
		non-repeaters	0	02	01	00
		max-repetitions	-1	02	01	***FF***
				30	00	
		variable-bindings				
			name			
			value			

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
		name			
		value			

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.11 GetBulk with negative non-repeaters, negative max-repetitions and zero variable bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with negative values for non-repeaters and max-repetitions, and zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

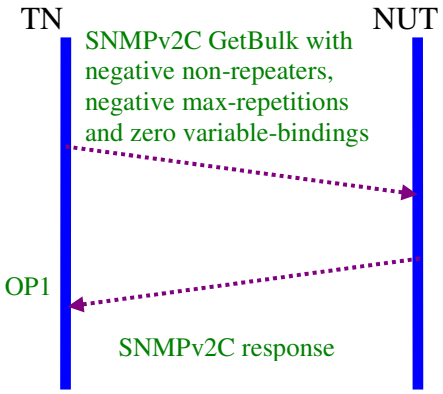
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulk with -1 for non-repeaters and max-repetitions and zero for variable-bindings.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	Version	1(SNMPv2C)	02	01	01

	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	0B	
		request-id	12	02	01	0C
		non-repeaters	-1	02	01	*FF*
		max-repetitions	-1	02	01	*FF*
				30	00	
		variable-bindings				
			name			
			value			

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
		name			
		value			

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.12 GetBulk with zero non-repeaters, negative max-repetitions and non-zero variable bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with zero non-repeaters, negative values for max-repetitions and non-zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

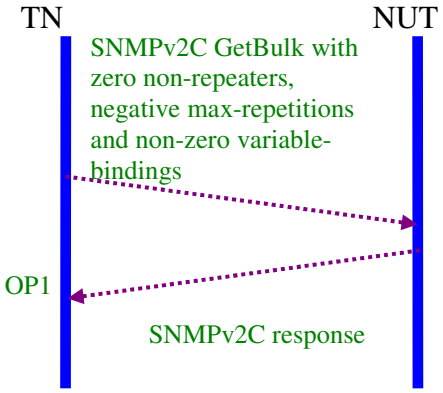
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN send SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with 0 for non-repeaters and -1 for max-repetitions and non zero variable-bindings.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	34	

	Version		1(SNMPv2C)	02	01	01
	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	27	
		request-id	12	02	01	0C
		non-repeaters	0	02	01	00
		max-repetitions	-1	02	01	*FF*
				30	18	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
			value	NULL	05	00	
			name				
			value				

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
	name				
	value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.13 GetBulk with negative non-repeaters, zero max-repetitions and non-zero variable bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with negative value for non-repeaters, zero max-repetitions and non-zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

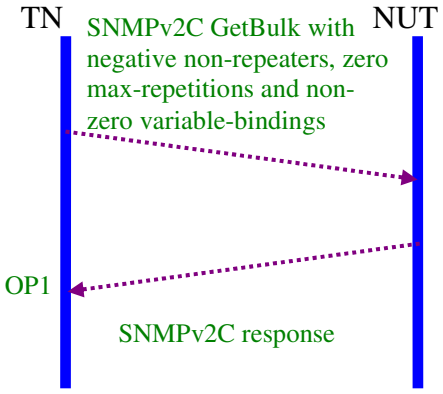
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest object to NUT by issuing SNMPv2C GetBulkRequest with -1 for non-repeaters, zero for max-repetitions, and non-zero variable-bindings.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	34	

	Version		1(SNMPv2C)	02	01	01
	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	27	
		request-id	12	02	01	0C
		non-repeaters	-1	02	01	***FF***
		max-repetitions	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
	name				
	value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.14 GetBulk with negative non-repeaters, negative max-repetitions and non-zero variable-bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with negative values for non-repeaters, max-repetitions and non-zero variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

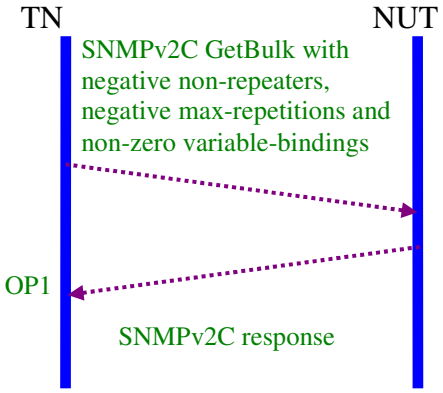
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with -1 for non-repeaters, max-repetitions and non zero variable-bindings.
2. NUT replies SNMPv2C Response with correct OID values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	34	

	Version		1(SNMPv2C)	02	01	01
	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	27	
		request-id	12	02	01	0C
		non-repeaters	-1	02	01	*FF*
		max-repetitions	-1	02	01	*FF*
				30	1C	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
		value	NULL	05	00	
		name				
		value				

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	18	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	0B	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	00	
	variable-bindings				
	name				
	value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with
1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be empty

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.15 GetBulk with large max-repetitions

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with large max-repetitions packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

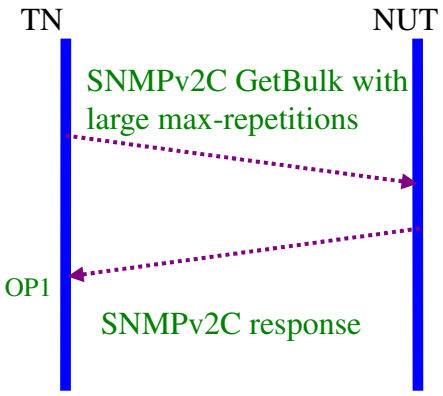
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with max-repetitions=100.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	25	
	Version	1(SNMPv2C)	02	01	01
	Community	public	04	06	70 75 62 6C 69 63

	D a t a	PDU type		GetBulkRequest	A5	18	
		request-id		12	02	01	0C
		non-repeaters		0	02	01	00
		max-repetitions		100	02	01	64
					30	0D	
		variable-bindings			30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	D a t a	PDU type	Response	A2	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value				*
					30	*	
			name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
			value				*
					30	*	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value				*
					30	*	***Repeat until 100 occurrences***
			name				
			value				

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding list should be the sysDescr OID and its value and the following 99 OID and their values in the MIB or End Of MIB is reached before the total the 100s variable-bindings is reached.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.16 GetBulk with non-repeaters greater than variable-bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with non-repeaters value greater than the number of variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

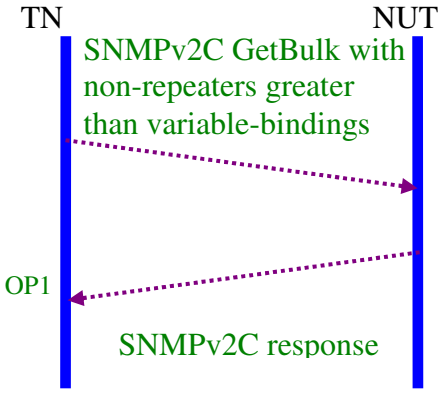
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with N=30 for non-repeaters and 10 sysDescr OIDs as the variable-bindings.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	819	

						C(156)	
	Version			1(SNMPv2C)	02	01	01
	Community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		GetBulkRequest	A5	818e (142)	
		request-id		12	02	01	0C
		non-repeaters		30	02	01	1E
		max-repetitions		0	02	01	0
					30	8182 (130)	
		variable-bindings			30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	

			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address				TN_ADDRESS		
	Destination Address				NUT_ADDRESS		
UDP Header	Source Port				161		
	Destination Port				Same as the source port in 1st packet		
SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	*	
	version			1(SNMPv2C)	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	Data	PDU type		Response	A2	*	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	*	
		variable-bindings			30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00

			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*
					30		
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value		04	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned variable binding list is 10 repetitions of 1.3.6.1.2.1.1.1.0(sysDescr.0) and its value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.17 GetBulk with non-repeaters less than variable-bindings

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with non-repeaters value less than the number of variable-bindings packet SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

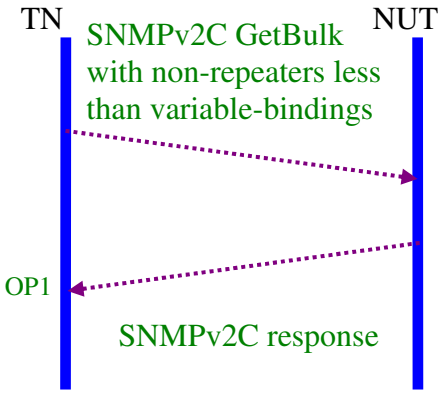
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with non-repeaters value less than the number of the variable-bindings.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	32	
	Version	1(SNMPv2C)	02	01	01

	Community			public	04	06	70 75 62 6C 69 63
	D	PDU type		GetBulkRequest	A5	25	
	a	request-id		12	02	01	0C
	t	non-repeaters		1	02	01	01
	a	max-repetitions		0	02	01	0
					30	0D	
		variable-bindings			30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	
					30	0B	
			name	1.3.6.1.2.1.1.1 (sysDescr)	06	07	2b 06 01 02 01 01 01
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address			TN_ADDRESS			
	Destination Address			NUT_ADDRESS			
UDP Header	Source Port			161			
	Destination Port			Same as the source port in 1st packet			
SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	18	
	version			1(SNMPv2C)	02	01	01
	community			public	04	06	70 75 62 6C 69 63
	D	PDU type		Response	A2	0B	
	a	request-id		12	02	01	0C
	t	error-status		0	02	01	00
	a	error-index		0	02	01	00
					30	00	
		var			30		

		iab le-bin din gs	name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	09	2b 06 01 02 01 01 01 01 00
			value		04	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned variable binding list should be1.3.6.1.2.1.1.1.0(sysDescr.0) and its values

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.18 GetBulk with 128 sub-identifiers

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object with 128 sub-identifiers from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

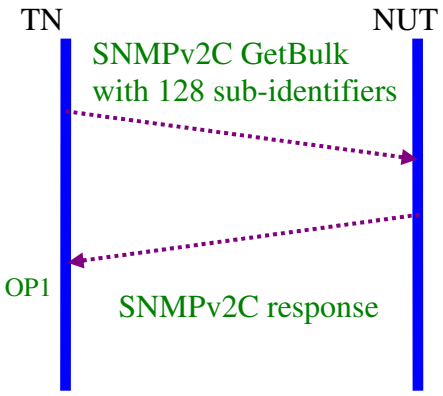
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest with 128 sub-ids to NUT.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	81a2 (162)	
	Version	1(SNMPv2C)	02	01	01

	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	8194 (148)	
		request-id	12	02	01	0C
		non-repeaters	1	02	01	1
		max-repetitions	0	02	01	0
				30	8188 (136)	
		variable-bindings		30	8185 (133)	
		name	1.3.6.1.2.1.1.1.3.4.5.6.7.8…..121.128 (128 sub IDs)	06	8180 (128)	2b 06 01 02 01 01 01 03 04 05 06 07 …..79 81 00
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	1(SNMPv2C)	02	01	01
	community	public	04	06	70 75 62 6C 69 63
	Data PDU type	Response	A2	*	
	Data request-id	12	02	01	0C
	Data error-status	0	02	01	00
	Data error-index	0	02	01	00
			30	*	
	variable-bindings		30	*	
	variable-bindings name	1.3.6.1.2.1.1.2.0 (sysObjectID.0)	06	08	2b 06 01 02 01 01 02 00
	variable-bindings value		06	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned variable binding is 1.3.6.1.2.1.1.2.0(sysObjectID.0) and its value

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.19 GetBulk with large Index ID

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest with large IID and sysDescr as the variable-bindings packet from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

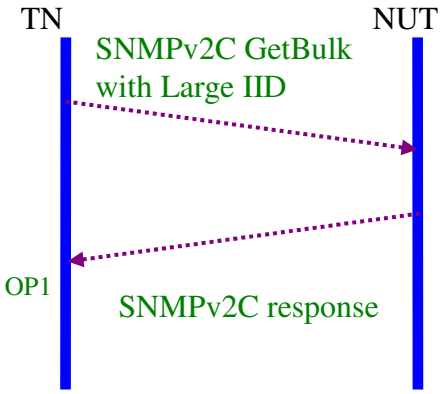
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest to NUT by issuing SNMPv2C GetBulkRequest with N=30 for non-repeaters and sysUpTime.4294967295 as variable-binding's name
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	2A	
	Version	1(SNMPv2C)	02	01	01

	Community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	GetBulkRequest	A5	29	
		request-id	12	02	01	0C
		non-repeaters	30	02	01	1E
		max-repetitions	0	02	01	0
				30	12	
		variable-bindings		30	10	
			name	1.3.6.1.2.1.1.3.4294967295 (sysUpTime.4294967295)	06	0C	2b 06 01 02 01 01 03 8F FF FF FF 7F
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Response	A2	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value		04	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the returned variable binding is 1.3.6.1.2.1.1.4.0(sysContact.0) and its value.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

v6SNMPv2C3.20 GetBulk with different tables

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the GetBulkRequest object from different tables from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

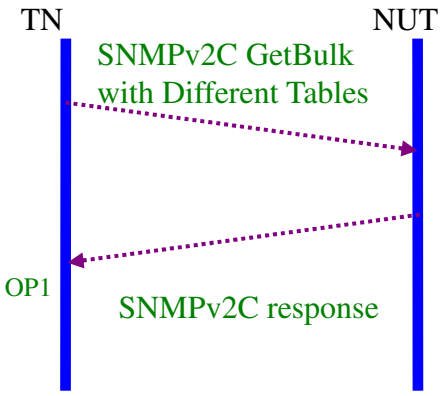
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetBulkRequest object to NUT by issuing SNMPv2C GetBulkRequest from ifTable table and udpTable.
2. NUT replies SNMPv2C Response with correct values to TN.

1st received packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	36	
	Version	1(SNMPv2C)	02	01	01
	Community	public	04	06	70 75 62 6C 69 63

	D a t a	PDU type		GetBulkRequest	A5	29	
		request-id		12	02	01	0C
		non-repeaters		2	02	01	02
		max-repetitions		0	02	01	00
					30	0E	
		variable-bindings			30	0D	
			name	1.3.6.1.2.1.2.2.1.1 (ifIndex)	06	09	2b 06 01 02 01 02 02 01 01
			value	NULL	05	00	
					30	0D	
			name	1.3.6.1.2.1.7.5.1.2 (udpLocalPort)	06	09	2b 06 01 02 01 07 05 01 02
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		public	04	06	70 75 62 6C 69 63
	D a t a	PDU type	Response	A2	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.2.2.1.1.[ifIndex]	06	*	2b 06 01 02 01 02 01 02 02 01 *
			value		02	*	*
					30	*	
			name	1.3.6.1.2.1.7.5.1.2.[udpLocalAddress].[udpLocalPort]	06	*	2b 06 01 02 01 07 05 01 02 *
			value		02	*	*

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetBulk request correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. the variable binding should be ifIndex.[ifIndex] and udpLocalPort.[udpLocalAddress].[udpLocalPort] and their values.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec.4.2.3

Group 4 IPv6 SNMPv2C SetRequest

Scope

The following tests verify the SetRequest commands in SNMPv2C protocol.

Overview

The SNMPv2C SetRequest-PDU is initiated by a SNMPv2C manager to set certain object value as defined in RFC1157. SNMPv2C agent, upon receiving such SetRequest-PDU, should reply with correct Response message after setting such object value in the MIB. The write community for this SetRequest test is private. The OID for any SetRequest operation should have read-write access mode for this SetRequest testing. Due to the fact that it might be difficult to perform this test with possible write mode privilege in some systems, this SetRequest test is optional for this SNMPv2C IPv6 Ready Logo testing. Please make sure to save the pre SetRequest variable values and perform a roll-back operation to restore MIB values for every successful SetRequest operation.

v6SNMPv2C4.1 Set non-existent object

Purpose

Verify that NUT playing the SNMPv2C agent can properly detect SetRequest on non-existent OID and will return Response PDU with the error-status field of noAccess or notWritable.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

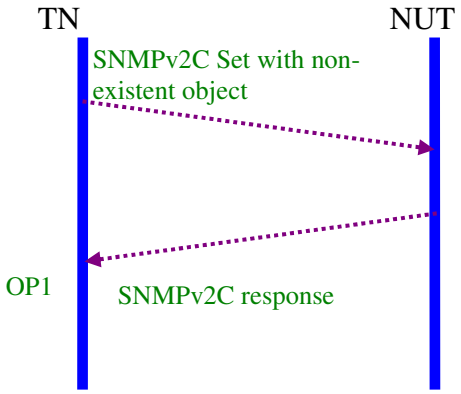
Please refer Fig. 5 Test Architecture.

Setup

Refer Fig. 6 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C SetRequest with non-existent object to NUT.
2. NUT replies SNMPv2C Response with correct error status code(notWritable or noAccess).

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	28	

	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	1A	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0F	
		variable-bindings		30	0D	
			name	1.3.6.1.8.1	06	05	2b 06 01 08 01
			value	test	04	04	74 65 73 74

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	28	
	version	SNMPv2C	02	01	01
	community	private	04	07	70 72 69 76 61 74 65
	Data PDU type	Response	A2	1A	
	request-id	12	02	01	0C
	error-status	***17 or 6***	***02***	***01***	***11(notWritable or noAccess,06)***
	error-index	***1***	***02***	***01***	***01***
			30	0F	
	variable-bindings		30	0D	
	name	1.3.6.1.8.1	06	05	2b 06 01 08 01
	value	test	04	04	74 65 73 74

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status is either 17(notWritable) or 6(noAccess) and error-index is 1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.2 Set existent read-write object

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the SetRequest PDU on read/write access object from the SNMPv2C manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

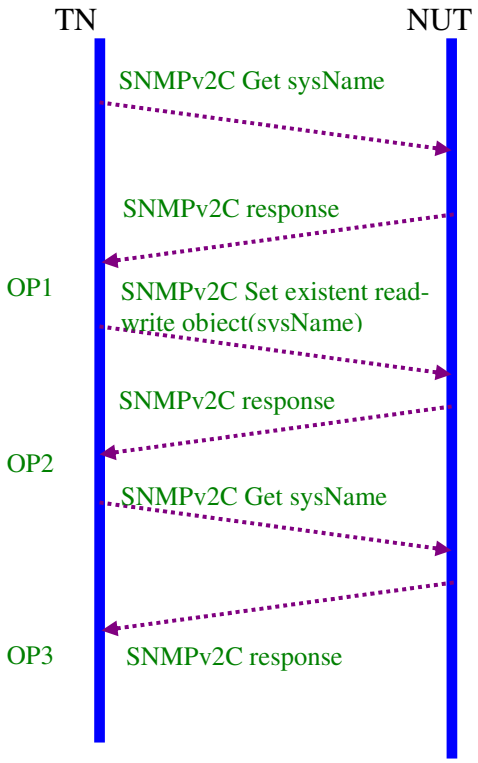
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C Get object to NUT by issuing SNMPv2C Get to check sysName.0 1.3.6.1.2.1.1.5.0 in system group in MIB II.
2. NUT replies SNMPv2C Get-response with correct values to TN. *Save the sysName value in this Response for rollback operation in 7.*
3. TN sends SNMPv2C SetRequest object to NUT by issuing SNMPv2C SetRequest to set sysName.0 1.3.6.1.2.1.1.5.0 in system group in MIB II

4. NUT replies SNMPv2C Response with correct sysName value to TN
5. TN sends SNMPv2C Get object to NUT by issuing SNMPv2C Get to verify the new sysName value
6. NUT replies SNMPv2C Response with correct sysName value to TN
7. *TN must perform another Set operation with the saved sysName in 3 to reinstate the sysName value before the successful set operation*

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	27	
	version		1(SNMPv2C)	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Get	A0	19	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	0E	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			NUT_ADDRESS		
	Destination Address			TN_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		1(SNMPv2C)	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	D	PDU type	Response	A2	*	

	a t a	request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	*	
		variable-bindings			30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	NUT system name value	04	*	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	33	
	version		1(SNMPv2C)	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	25	
		request-id	13	02	01	0D
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1A	
		variable-bindings		30	18	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	publicpublic	04	0C	70 75 62 6C 69 63 70 75 62 6C 69 63

4th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

				30	33	
	version		1(SNMPv2C)	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	25	
		request-id	13	02	01	OD
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1A	
		variable-bindings		30	18	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	***publicpublic***	***04***	***0C***	***70 75 62 6C 69 63 70 75 62 6C 69 63***

5th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	27	
	version	1(SNMPv2C)	02	01	01
	community	private	04	07	70 72 69 76 61 74 65
	Data PDU type	Get	A0	19	
	request-id	14	02	01	0E
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	0E	
	variable-bindings		30	0C	
	name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
	value	NULL	05	00	

6th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)		
IP Header	Source Address	NUT_ADDRESS
	Destination Address	TN_ADDRESS
UDP	Source Port	161

Header	Destination Port			Same as the 5th packet source port		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	33	
	version		1(SNMPv2C)	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	25	
		request-id	14	02	01	0E
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1A	
		variable-bindings		30	18	
		name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
		value	***publicpublic***	***04***	***0C***	***70 75 62 6C 69 63 70 75 62 6C 69 63***

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C Get sysName request correctly. This packet is with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. value field is the system name in system group of NUT with correct syntax type and value within the defined range field

OP2: TN received SNMPv2C Response from NUT after sending SNMPv2C SetRequest command. The packet received is with

5. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
6. request-id is the same as the request id in the previously received SNMPv2C SetRequest
7. error-status must be equal to zero and error-index must be equal zero
8. value field is the new system name in system group of NUT with correct syntax type and value

OP3: TN received SNMPv2C Response from NUT after sending SNMPv2C Get command. The packet received is with

9. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
10. request-id is the same as the request id in the previously received SNMPv2C GetRequest
11. error-status must be equal to zero and error-index must be equal zero
12. value field is **TRULY** the new system name in system group of NUT with correct syntax type and value

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.3 Set existent read-write object error
v6SNMPv2C4.3.1 Set with wrongType

Purpose

Verify that NUT playing the SNMPv2C agent can properly detect error of writable variable's value and will return Response PDU with the error-status field of wrongType.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

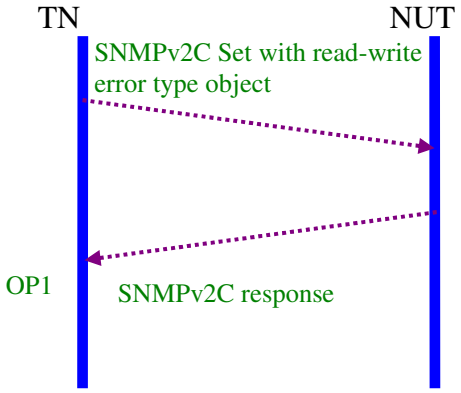
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 6 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C SetRequest PDU to set NUT's read/write object(sysName.0) but with error type of value.
2. NUT replies SNMPv2C Response with wrongType.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

				30	2B	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	1D	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	12	
		variable-bindings		30	10	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	test	**02**	04	74 65 73 74

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	2B	
	version	SNMPv2C	02	01	01
	community	private	04	07	70 72 69 76 61 74 65
	Data PDU type	Response	A2	1D	
	request-id	12	02	01	0C
	error-status	7	02	01	7(WrongType)
	error-index	1	02	01	01
			30	12	
	variable-bindings		30	10	
	name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
	value	test	02	04	74 65 73 74

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status is wrongType and error-index is 1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.3.2 Set with wrongValue

Purpose

Verify that NUT playing the SNMPv2C agent can properly detect invalid value in writing a string variable and will return Response PDU with the error-status field of wrongValue.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

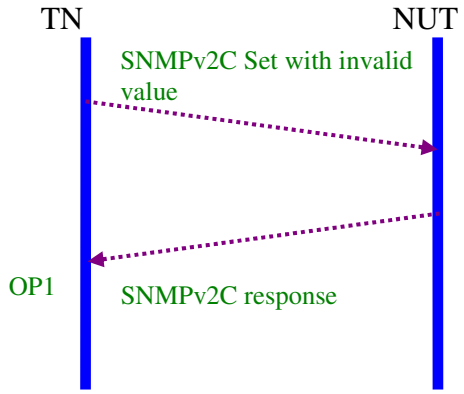
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 3 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C SetRequest with invalid value to NUT.
2. NUT replies SNMPv2C Response with wrongValue.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	28	
	version	SNMPv2C	02	01	01
	community	private	04	07	70 72 69 76 61 74

						65
	D a t a	PDU type		SetRequest	A3	1A	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	0F	
		variable-bindings			30	0D	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	?	04	01	FF

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	28	
	version	SNMPv2C	02	01	01
	community	private	04	07	70 72 69 76 61 74 65
	D a t a PDU type	Response	A2	1A	
	request-id	12	02	01	0C
	error-status	10	02	01	0A(wrongValue)
	error-index	1	02	01	01
			30	0F	
	variable-bindings		30	0D	
	name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
	value	?	04	01	FF

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status is 10(wrongValue) and error-index is 1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.3.3 Set existent read-write object with non-existent instance

Purpose

Verify that NUT playing the SNMPv2C agent can properly detect SetRequest PDU setting non-existent instance in the relevant MIB from SNMPv2C manager and will return Response PDU with the error-status field of 17(notWritable) or 6(noAccess) from NUT

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

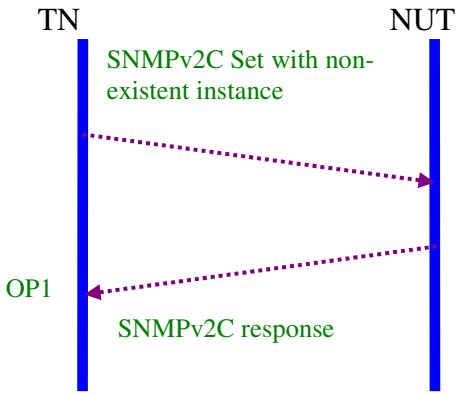
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 6 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C SetRequest read-write MIB object but with non-existent instance to NUT.
2. NUT replies SNMPv2C Response with notWritable or noAccess.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value

				30	2B	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	1D	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	12	
		variable-bindings		30	10	
			name	1.3.6.1.2.1.1.5.100	06	08	2b 06 01 02 01 01 05 64
			value	test	04	04	74 65 73 74

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	2B	
	version	SNMPv2C	02	01	01
	community	private	04	07	70 72 69 76 61 74 65
	Data PDU type	Response	A2	1D	
	request-id	12	02	01	0C
	error-status	17 or 6	02	01	11(notWritable) or 6(noAccess,06)
	error-index	1	02	01	01
			30	12	
	variable-bindings		30	10	
		name	1.3.6.1.2.1.1.5.100	06	08	2b 06 01 02 01 01 05 64
		value	test	04	04	74 65 73 74

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status is 17(notWritable) or 6(noAccess) and error-index is 1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.4 Set existent read-only object with existent instance

Purpose

Verify that NUT playing the SNMPv2C agent can properly process the SetRequest PDU setting read-only instance in the relevant MIB from SNMPv2C manager and will return Response PDU with error status of 17(notWritable).

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

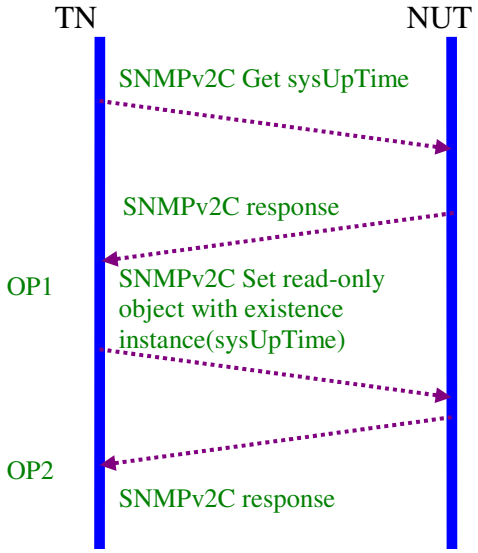
Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest object to NUT by issuing SNMPv2C Get to check sysUpTime (1.3.6.1.2.1.1.3.0) in system group in MIB II.
2. NUT replies SNMPv2C Response with correct values to TN.
3. TN sends SNMPv2C SetRequest to NUT by issuing SNMPv2C SetRequest to set sysUpTime 1.3.6.1.2.1.1.3.0 in system group in MIB II.
4. NUT replies SNMPv2C Response with correct value to TN.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS

<table>
<tr><td rowspan="2">UDP Header</td><td colspan="3">Source Port</td><td colspan="3">any</td></tr>
<tr><td colspan="3">Destination Port</td><td colspan="3">161</td></tr>
<tr><td rowspan="11">SNMP Message</td><td colspan="3" rowspan="2">SNMP Fields</td><td rowspan="2">Values (readable)</td><td colspan="3">ASN.1(Hex)</td></tr>
<tr><td>type</td><td>len</td><td>value</td></tr>
<tr><td colspan="3"></td><td></td><td>30</td><td>27</td><td></td></tr>
<tr><td colspan="3">version</td><td>1(SNMPv2C)</td><td>02</td><td>01</td><td>01</td></tr>
<tr><td colspan="3">community</td><td>private</td><td>04</td><td>07</td><td>70 72 69 76 61 74 65</td></tr>
<tr><td rowspan="9">Data</td><td colspan="2">PDU type</td><td>Get</td><td>A0</td><td>19</td><td></td></tr>
<tr><td colspan="2">request-id</td><td>12</td><td>02</td><td>01</td><td>0C</td></tr>
<tr><td colspan="2">error-status</td><td>0</td><td>02</td><td>01</td><td>00</td></tr>
<tr><td colspan="2">error-index</td><td>0</td><td>02</td><td>01</td><td>00</td></tr>
<tr><td colspan="2"></td><td></td><td>30</td><td>OE</td><td></td></tr>
<tr><td rowspan="3">variable-bindings</td><td></td><td></td><td>30</td><td>0C</td><td></td></tr>
<tr><td>name</td><td>1.3.6.1.2.1.1.3.0 (sysUpTime.0)</td><td>06</td><td>08</td><td>2b 06 01 02 01 01 03 00</td></tr>
<tr><td>value</td><td>NULL</td><td>05</td><td>00</td><td></td></tr>
</table>

2nd Packet

<table>
<tr><td colspan="8">Standard query response from SNMP agent (NUT) to SNMP manager (TN)</td></tr>
<tr><td rowspan="2">IP Header</td><td colspan="3">Source Address</td><td colspan="4">NUT_ADDRESS</td></tr>
<tr><td colspan="3">Destination Address</td><td colspan="4">TN_ADDRESS</td></tr>
<tr><td rowspan="2">UDP Header</td><td colspan="3">Source Port</td><td colspan="4">161</td></tr>
<tr><td colspan="3">Destination Port</td><td colspan="4">Same as the source port in 1st packet</td></tr>
<tr><td rowspan="11">SNMP Message</td><td colspan="3" rowspan="2">SNMP Fields</td><td rowspan="2">Values (readable)</td><td colspan="3">ASN.1(Hex)</td></tr>
<tr><td>type</td><td>len</td><td>value</td></tr>
<tr><td colspan="3"></td><td></td><td>30</td><td>*</td><td></td></tr>
<tr><td colspan="3">version</td><td>1(SNMPv2C)</td><td>02</td><td>01</td><td>01</td></tr>
<tr><td colspan="3">community</td><td>private</td><td>04</td><td>07</td><td>70 72 69 76 61 74 65</td></tr>
<tr><td rowspan="9">Data</td><td colspan="2">PDU type</td><td>Response</td><td>A2</td><td>*</td><td></td></tr>
<tr><td colspan="2">request-id</td><td>12</td><td>02</td><td>01</td><td>0C</td></tr>
<tr><td colspan="2">error-status</td><td>0</td><td>02</td><td>01</td><td>00</td></tr>
<tr><td colspan="2">error-index</td><td>0</td><td>02</td><td>01</td><td>00</td></tr>
<tr><td colspan="2"></td><td></td><td>30</td><td>*</td><td></td></tr>
<tr><td rowspan="3">variable-bindings</td><td></td><td></td><td>30</td><td>*</td><td></td></tr>
<tr><td>name</td><td>1.3.6.1.2.1.1.3.0 (sysUpTime.0)</td><td>06</td><td>08</td><td>2b 06 01 02 01 01 03 00</td></tr>
<tr><td>value</td><td>NUT's sysUpTime</td><td>43</td><td>*</td><td>*</td></tr>
</table>

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)							
IP Header	Source Address			TN_ADDRESS			
	Destination Address			NUT_ADDRESS			
UDP Header	Source Port			any			
	Destination Port			161			
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)			
				type	len	value	
				30	2B		
	version		1(SNMPv2C)	02	01	01	
	community		private	04	07	70 72 69 76 61 74 65	
	Data	PDU type	SetRequest	A3	1D		
		request-id	13	02	01	0D	
		error-status	0	02	01	00	
		error-index	0	02	01	00	
				30	12		
		variable-bindings		30	10		
			name	1.3.6.1.2.1.1.3.0 (sysUpTime)	06	08	2b 06 01 02 01 01 03 00
			value	129183270	43	04	07 b3 2e 26

4th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address			NUT_ADDRESS			
	Destination Address			TN_ADDRESS			
UDP Header	Source Port			161			
	Destination Port			Same as the 3rd packet source port			
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)			
				type	len	value	
				30	2B		
	version		1(SNMPv2C)	02	01	01	
	community		private	04	07	70 72 69 76 61 74 65	
	Data	PDU type	Response	A2	1D		
		request-id	13	02	01	0D	
		error-status	17	02	01	11(Not Writable)	
		error-index	1	02	01	1	
				30	12		
		variab		30	10		
			name	1.3.6.1.2.1.1.3.0	06	08	2b 06 01 02 01 01

		le-bindings		(sysUpTime.0)			03 00
			value	129183270	43	04	07 b3 2e 26

Note * indicates variable values that vary according with the actual packet and OIDs

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status must be equal to zero and error-index must be equal zero
4. variable's value field is the current system upTime in system group of NUT with correct syntax type and value within the defined range

OP2: TN received SNMPv2C Response from NUT after sending SNMPv2C SetRequest command

5. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
6. request-id is the same as the request id in the previously received SNMPv2C SetRequest
7. error-status is notWritable and error-index is set to 1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.5 Set multiple variables

v6SNMPv2C4.5.1 Set two read-write variables

Purpose

Verify that NUT playing the SNMPv2C agent can properly process SetRequest PDU setting on multiple different read/write variable instances from SNMPv2C manger and will return Response PDU correctly.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

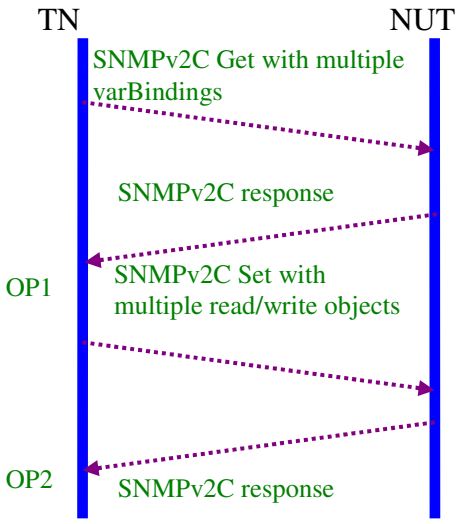
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 3 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with multiple variable-bindings to NUT to save original values before the SetRequest operation which will be restored afterwards.
2. NUT replies SNMPv2C Response with current variable values before SetRequest
3. TN sends SNMPv2C SetRequest object to NTU to set sysContact.0(1.3.6.1.2.1.1.4.0) and sysName.0(1.3.6.1.2.1.1.5.0) in system Group in MIB II.
4. NUT replies SNMPv2C Response with correct values to TN

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)

<table>
<tr><td rowspan="2">IP Header</td><td colspan="3">Source Address</td><td colspan="4">TN_ADDRESS</td></tr>
<tr><td colspan="3">Destination Address</td><td colspan="4">NUT_ADDRESS</td></tr>
<tr><td rowspan="2">UDP Header</td><td colspan="3">Source Port</td><td colspan="4">any</td></tr>
<tr><td colspan="3">Destination Port</td><td colspan="4">161</td></tr>
<tr><td rowspan="15">SNMP Message</td><td colspan="3" rowspan="2">SNMP Fields</td><td rowspan="2">Values (readable)</td><td colspan="3">ASN.1(Hex)</td></tr>
<tr><td>type</td><td>len</td><td>value</td></tr>
<tr><td colspan="3"></td><td></td><td>30</td><td>*</td><td></td></tr>
<tr><td colspan="3">version</td><td>SNMPv2C</td><td>02</td><td>01</td><td>01</td></tr>
<tr><td colspan="3">community</td><td>private</td><td>04</td><td>07</td><td>70 72 69 76 61 74 65</td></tr>
<tr><td rowspan="13">Data</td><td colspan="2">PDU type</td><td>GetRequest</td><td>A1</td><td>*</td><td></td></tr>
<tr><td colspan="2">request-id</td><td>12</td><td>02</td><td>01</td><td>0C</td></tr>
<tr><td colspan="2">error-status</td><td>0</td><td>02</td><td>01</td><td>00</td></tr>
<tr><td colspan="2">error-index</td><td>0</td><td>02</td><td>01</td><td>00</td></tr>
<tr><td colspan="2"></td><td></td><td>30</td><td>*</td><td></td></tr>
<tr><td rowspan="8">variable-bindings</td><td></td><td></td><td>30</td><td>*</td><td></td></tr>
<tr><td>name</td><td>1.3.6.1.2.1.1.4.0 (sysContact.0)</td><td>06</td><td>08</td><td>2b 06 01 02 01 01 04 00</td></tr>
<tr><td>value</td><td>NULL</td><td>05</td><td>00</td><td>00</td></tr>
<tr><td></td><td></td><td>30</td><td>0C</td><td></td></tr>
<tr><td>name</td><td>1.3.6.1.2.1.1.5.0 (sysName.0)</td><td>06</td><td>08</td><td>2b 06 01 02 01 01 05 00</td></tr>
<tr><td>value</td><td>NULL</td><td>05</td><td>0</td><td>0</td></tr>
</table>

2nd Packet

<table>
<tr><td colspan="8">Standard query response from SNMP agent (NUT) to SNMP manager (TN)</td></tr>
<tr><td rowspan="2">IP Header</td><td colspan="3">Source Address</td><td colspan="4">TN_ADDRESS</td></tr>
<tr><td colspan="3">Destination Address</td><td colspan="4">NUT_ADDRESS</td></tr>
<tr><td rowspan="2">UDP Header</td><td colspan="3">Source Port</td><td colspan="4">161</td></tr>
<tr><td colspan="3">Destination Port</td><td colspan="4">Same as the source port in 1st packet</td></tr>
<tr><td rowspan="14">SNMP Message</td><td colspan="3" rowspan="2">SNMP Fields</td><td rowspan="2">Values (readable)</td><td colspan="3">ASN.1(Hex)</td></tr>
<tr><td>type</td><td>len</td><td>value</td></tr>
<tr><td colspan="3"></td><td></td><td>30</td><td>3A</td><td></td></tr>
<tr><td colspan="3">version</td><td>SNMPv2C</td><td>02</td><td>01</td><td>01</td></tr>
<tr><td colspan="3">community</td><td>private</td><td>04</td><td>07</td><td>70 72 69 76 61 74 65</td></tr>
<tr><td rowspan="12">Data</td><td colspan="2">PDU type</td><td>Response</td><td>A2</td><td>2C</td><td></td></tr>
<tr><td colspan="2">request-id</td><td>12</td><td>02</td><td>01</td><td>0C</td></tr>
<tr><td colspan="2">error-status</td><td>0</td><td>02</td><td>01</td><td>0</td></tr>
<tr><td colspan="2">error-index</td><td>0</td><td>02</td><td>01</td><td>0</td></tr>
<tr><td colspan="2"></td><td></td><td>30</td><td>21</td><td></td></tr>
<tr><td rowspan="2">variab</td><td></td><td></td><td>30</td><td>10</td><td></td></tr>
<tr><td></td><td>1.3.6.1.2.1.1.4.0</td><td>06</td><td>08</td><td>2b 06 01 02 01 01</td></tr>
</table>

		le-bindings		sysContact.0			04 00
				*	04	*	*
					30	0D	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	*	04	*	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	3A	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	2C	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	21	
		variable-bindings		30	10	
			name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value	ipv6	04	04	69 70 76 36
					30	0D	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	test	04	04	74 65 73 74

4th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	3A	
	version	SNMPv2C	02	01	01

	community			private	04	07	70 72 69 76 61 74 65
	D a t a	PDU type		Response	A2	2C	
		request-id		12	02	01	0C
		error-status		0	02	01	0
		error-index		0	02	01	0
					30	21	
		var iab le- bin din gs			30	10	
			name	1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
			value	ipv6	04	04	69 70 76 36
					30	0D	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	test	04	04	74 65 73 74

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with
1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status and error-index must be equal to 0
4. two instances must be in the variable binding list with their correct syntax types and values within their defined range

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with
5. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
6. request-id is the same as the request id in the previously received SNMPv2C SetRequest
7. error-status and error-index must be equal to 0
8. two instances must be in the variable binding list with their correct syntax types and values within their defined range

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.5.2 Set two read-write variables with wrong type of the second variable

Purpose

Verify that NUT playing the SNMPv2C agent can properly detect SetRequest PDU setting on multiple read/write access variables but with wrong type of value(the first instance) from SNMPv2C manager and will return Response PDU correctly.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

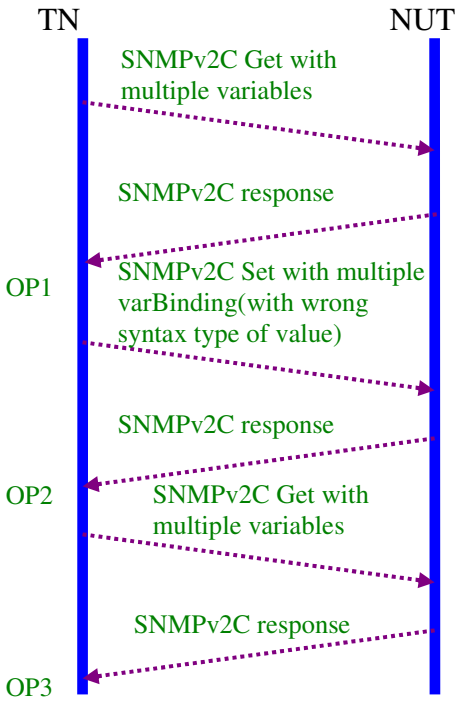
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 3 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with multiple variables to NUT to save original values before SetRequest operation which will be restored afterwards.
2. NUT replies SNMPv2C Response with the current variables values before SetRequest
3. TN sends SNMPv2C SetRequest to NUT to set sysContact.0(1.3.6.1.2.1.1.4.0) and sysName.0(1.3.6.1.2.1.1.5.0) but with wrong syntax type of value in sysContact in MIB II system group.

4. NUT replies SNMPv2C Response with corresponding error-status and error-index.
5.TN sends SNMPv2C GetRequest with multiple variables to NUT to check if the values have seen correctly set after the Set operation
6.NUT replies SNMPv2C Response with current values

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	35	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			value	NULL	05	00	
				30	0C	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		161		
	Destination Port		Same as the source port in 1st packet		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	SNMPv2C	02	01	01
	community	private	04	07	70 72 69 76 61 74 65

	D a t a	PDU type		Response	A2	*	
		request-id		12	02	01	0C
		error-status		0	02	01	0
		error-index		0	02	01	0
					30	21	
		variable-bindings			30	*	
			name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			value	*	04	*	*
					30	0C	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	*	04	*	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		TN_ADDRESS		
	Destination Address		NUT_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		161		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	3A	
	version	SNMPv2C	02	01	01
	community	private	04	07	70 72 69 76 61 74 65
	D a t a PDU type	SetRequest	A3	2C	
	request-id	12	02	01	0C
	error-status	0	02	01	00
	error-index	0	02	01	00
			30	21	
	variable-bindings		30	10	
	name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
	value	ipv6	04	04	69 70 76 36
			30	0D	
	name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
	value	100	02	01	64

4th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP	Source Port	161

Header	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	3A	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	2C	
		request-id	12	02	01	0C
		error-status	***7***	***02***	***01***	***07(wrongType)***
		error-index	***2***	***02***	***01***	***02***
				30	21	
		variable-bindings		30	10	
			1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			ipv6	04	04	69 70 76 36
				30	0D	
		name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
		value	100	02	01	64

5th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	35	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.5.0	06	08	2b 06 01 02 01 01

				sysName.0			05 00
			value	NULL	05	00	

6th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)							
IP Header	Source Address			TN_ADDRESS			
	Destination Address			NUT_ADDRESS			
UDP Header	Source Port			any			
	Destination Port			161			
SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	*	
	version			SNMPv2C	02	01	01
	community			private	04	07	70 72 69 76 61 74 65
	Data	PDU type		Response	A2	*	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	*	
		variable-bindings			30	10	
			name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			value	ipv6	04	04	69 70 76 36
					30	*	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	*	04	*	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest scalar correctly.

Received packet with
1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C GetRequest
3. error-status and error-index is 0
4. two instances must be in the variable binding list with their correct syntax types and values within their defined range

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

5. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
6. request-id is the same as the request id in the previously received SNMPv2C SetRequest
7. error-status is wrongValue and error-index is 2
8. the first set OID is correctly set while the second OID is not set

OP3: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with

9. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
10. request-id is the same as the request id in the previously received SNMPv2C GetRequest
11. error-status and error-index must be equal to zero
12. the first variable is correctly set while the second sysName.0 is not set(match the values in OP1)

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.5.3 Set two read-write variables with wrong type of the first variable

Purpose

Verify that NUT playing the SNMPv2C agent can properly detect wrong variable-binding correctly and respond with correct error-index code.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

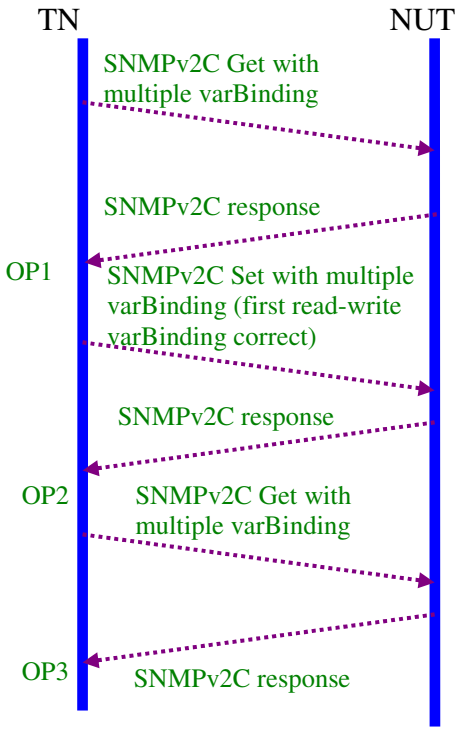
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 3 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with multiple variable-bindings to NUT to save pre SetRequest OID value which will be restored afterwards.
2. NUT replies SNMPv2C Response with before the SetRequest
3. TN sends SNMPv2C SetRequest with multiple variable-bindings to NUT.
4. NUT replies SNMPv2C Response with correct error-status and error-index
5. TN sends SNMPv2C GetRequest with multiple variables to NUT to check if the

values have seen correctly set after the Set operation
6.NUT replies SNMPv2C Response with current values

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	35	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			value	NULL	05	00	
				30	0C	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	3A	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	D	PDU type	Response	A2	2C	

	a t a	request-id		12	02	01	0C
		error-status		0	02	01	0
		error-index		0	02	01	0
					30	21	
		variable-bindings			30	10	
				1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
				*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	*	04	*	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	3A(58)	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	2C(44)	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	21(33)	
		variable-bindings		30	0D	
			name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			value	**100**	**02**	**01**	**64**
					30	10	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	test	04	04	74 65 73 74

4th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)		
IP Header	Source Address	TN_ADDRESS

	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	3A	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	2C	
		request-id	12	02	01	0C
		error-status	***7***	***02***	***01***	***07(wrongType)***
		error-index	***1***	***02***	***01***	***01***
				30	21	
		variable-bindings		30	0D	
			1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			100	02	01	64
				30	10	
		name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
		value	test	04	04	74 65 73 74

5th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	35	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bin		30	0C	
		name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
		value	NULL	05	00	

					30	0C	
		bindings	name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	NULL	05	00	

6th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	10	
			name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			value	*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	test	04	04	74 65 73 74

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest scalar correctly.

Received packet with
1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status and error-index is 0

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status is wrongType(07) and error-index is 1

OP3: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status and error-index are equal to zero
4. Two instances must be in the variable binding list with their correct syntax types and values within the defined range field and match the values OP1 stored.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.5.4 Set two read-write variables with wrong type of the variables

Purpose

Verify that NUT playing the SNMPv2C agent can properly detect SetRequest PDU setting on multiple read/write access rights variables but with wrong syntax type of value for each variable.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

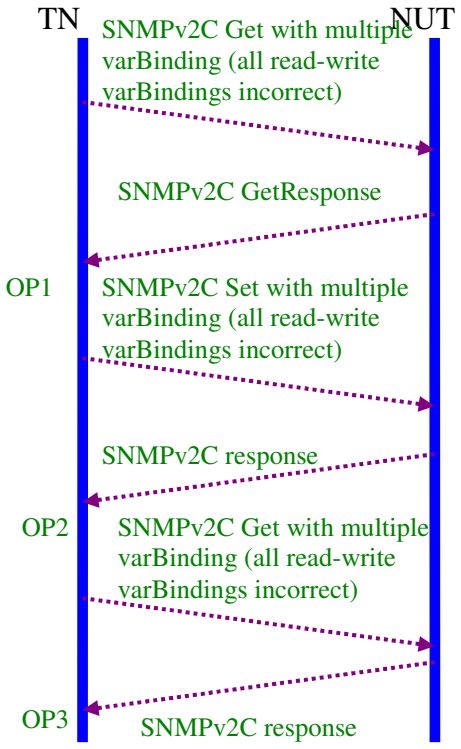
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 3 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with multiple variable-bindings to NUT to save pre SetRequest OID value which will be restored afterwards.
2. NUT replies SNMPv2C Response before the SetRequest
3. TN sends SNMPv2C SetRequest with multiple variable-bindings to NUT.
4. NUT replies SNMPv2C Response with correct error code

5. TN sends SNMPv2C Get Request with multiple variables to NUT to check if the variables are correctly set by the Set operation
6.NUT replies SNMPv2C Response with current values

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	34	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.4.0 (sysConatct.0)	06	08	2b 06 01 02 01 01 04 00
			value	NULL	05	00	
				30	0C	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	39	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	D	PDU type	Response	A2	2C	

	a t a	request-id		12	02	01	0C
		error-status		0	02	01	0
		error-index		0	02	01	0
					30	21	
		variable-bindings			30	10	
				1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
				*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	*	04	*	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	37	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	29	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1E	
		variable-bindings		30	0D	
			name	1.3.6.1.2.1.1.4.0 (sysConact.0)	06	08	2b 06 01 02 01 01 04 00
			value	***100***	***02***	***01***	***64***
					30	0D	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	***101***	***02***	***01***	***65***

4th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	any
	Destination Port	161

SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	37	
	version			SNMPv2C	02	01	01
	community			private	04	07	70 72 69 76 61 74 65
	Data	PDU type		Response	A2	29	
		request-id		12	02	01	0C
		error-status		7	02	01	07
		error-index		***1***	***02***	***01***	***01***
					30	1E	
		variable-bindings			30	0D	
			name	1.3.6.1.2.1.1.4.0 sysContact.0	06	08	2b 06 01 02 01 01 04 00
			value	***100***	***02***	***01***	***64***
					30	0D	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	***101***	***02***	***01***	***65***

5th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	34	
	version			SNMPv2C	02	01	01
	community			private	04	07	70 72 69 76 61 74 65
	Data	PDU type		GetRequest	A1	27	
		request-id		12	02	01	0C
		error-status		0	02	01	00
		error-index		0	02	01	00
					30	1C	
		variable-bindings			30	0C	
			name	1.3.6.1.2.1.1.4.0 (sysConatct.0)	06	08	2b 06 01 02 01 01 04 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	NULL	05	00	

6th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address				TN_ADDRESS		
	Destination Address				NUT_ADDRESS		
UDP Header	Source Port				161		
	Destination Port				Same as the source port in 1st packet		
SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	39	
	version			SNMPv2C	02	01	01
	community			private	04	07	70 72 69 76 61 74 65
	Data	PDU type		Response	A2	2C	
		request-id		12	02	01	0C
		error-status		0	02	01	0
		error-index		0	02	01	0
					30	21	
		variable-bindings			30	10	
				1.3.6.1.2.1.1.4.0 (sysContact.0)	06	08	2b 06 01 02 01 01 04 00
				*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	*	04	*	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest scalar correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status and error-index must be zero
4. two instances must be in the variable binding list with their correct syntax types and values within their defined range

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C

SetRequest correctly.

Received packet with

5. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
6. request-id is the same as the request id in the previously received SNMPv2C SetRequest
7. error-status is wrongType and error-index is 1

OP3: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with

8. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
9. request-id is the same as the request id in the previously received SNMPv2C GetRequest
10. error-status and error-index must be zero
11. variable-bindings list must match the values received in OP1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.5.5 Set read-write and read-only variables

Purpose

Verify that NUT playing the SNMPv2C agent can properly detect SetRequest PDU setting on hybrid read/write and read-only access mode variables from SNMPv2C manager and will return Response PDU correctly.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

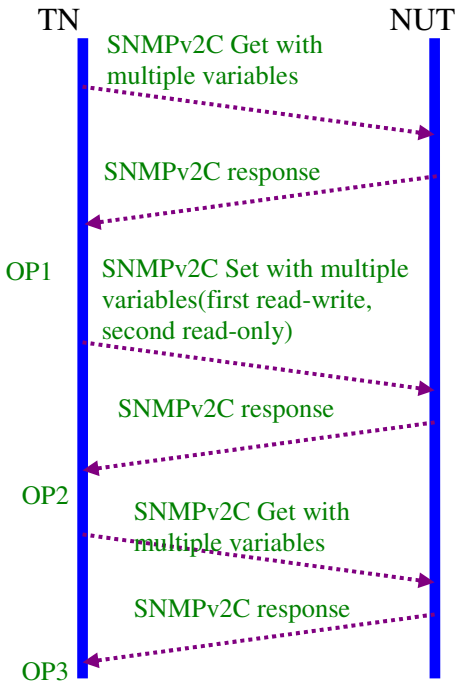
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 3 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with multiple variables to NUT to save pre SetRequest OID value which will be restored afterwards.
2. NUT replies SNMPv2C Response before the SetRequest
3. TN sends SNMPv2C SetRequest with multiple variables to NUT.
4. NUT replies SNMPv2C Response with correct error code.
5. TN sends SNMPv2C Get Request with multiple variables to NUT to check if the variables are correctly set by the Set operation

6.NUT replies SNMPv2C Response with current values

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	34	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	39	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	2C	
		request-id	12	02	01	0C
		error-status	0	02	01	0
		error-index	0	02	01	0

					30	21	
		variable-bindings			30	10	
				1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
				*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	*	04	*	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	10	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	**test**	**04**	**04**	**74 65 73 74**
					30	0D	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	***snmp_agent***	***04***	***0A***	***73 6e 6d 70 5F 61 67 65 6E 74***

4th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		161
	Destination Port		Same as the source port in 1st packet
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	*	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	*	
		request-id	12	02	01	0C
		error-status	17	02	01	11
		error-index	***2***	02	01	***02***
				30	*	
		variable-bindings		30	10	
			1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			test	04	04	74 65 73 74
				30	0D	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	snmp_agent	04	0A	73 6e 6d 70 5F 61 67 65 6E 74

5th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	34	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	NULL	05	00	

6th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address			TN_ADDRESS			
	Destination Address			NUT_ADDRESS			
UDP Header	Source Port			161			
	Destination Port			Same as the source port in 1st packet			
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)			
				type	len	value	
				30	39		
	version		SNMPv2C	02	01	01	
	community		private	04	07	70 72 69 76 61 74 65	
	Data	PDU type	Response	A2	2C		
		request-id	12	02	01	0C	
		error-status	0	02	01	0	
		error-index	0	02	01	0	
				30	21		
		variable-bindings		30	10		
			1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00	
			*	04	*	*	
				30	*		
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00	
		value	*	04	*	*	

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest scalar correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status and error-index is 0
4. two instances must be in the variable binding list with their correct syntax types and values within their defined range

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

5. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
6. request-id is the same as the request id in the previously received SNMPv2C SetRequest
7. error-status is notWritable and error-index is 2

OP3: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with

8. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
9. request-id is the same as the request id in the previously received SNMPv2C GetRequest
10. error-status and error-index are equal to zero
11. variable-bindings list must match the values received in OP1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.5.6 Set read-write variable with wrong type and read-only variable

Purpose

Verify that NUT playing the SNMPv2C agent can properly process SetRequest setting on hybrid read/write (with wrong syntax type of value) and read-only variables from SNMPv2C manger and will return Response PDU correctly.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

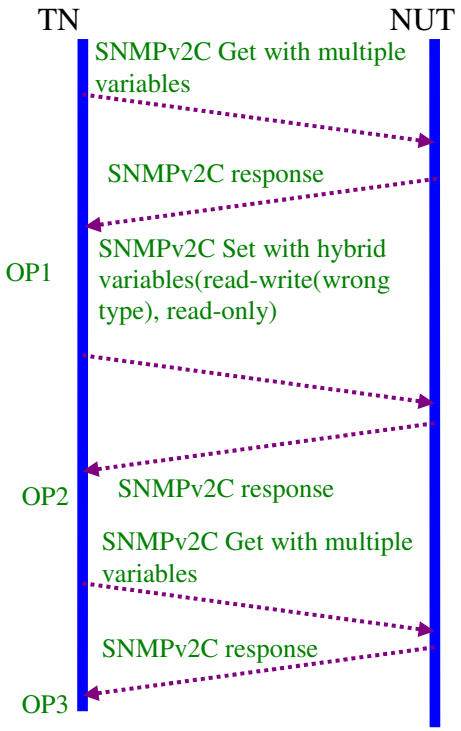
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 3 Common IPv6 Link Test Setup Basic Before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with multiple variables to NUT to save pre SetRequest OID value which will be restored afterwards.
2. NUT replies SNMPv2C Response before the SetRequest
3. TN sends SNMPv2C SetRequest with multiple variables to NUT.
4. NUT replies SNMPv2C Response with correct error code

5. TN sends SNMPv2C Get Request with multiple variables to NUT to check the variables after the Set operation
6.NUT replies SNMPv2C Response with current values

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	34	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	NULL	05	00	
				30	0C	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	39	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	D	PDU type	Response	A2	2C	

	a t a	request-id		12	02	01	0C
		error-status		0	02	01	0
		error-index		0	02	01	0
					30	21	
		variable-bindings			30	10	
				1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
				*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	*	02	01	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	36	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	29	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1E	
		variable-bindings		30	0D	
			name	1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			value	***100***	***02***	***01***	***64***
					30	*	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	*	02	*	*

4th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)		
IP Header	Source Address	TN_ADDRESS
	Destination Address	NUT_ADDRESS
UDP Header	Source Port	161
	Destination Port	Same as the source port in 1st

				packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	*	
		request-id	12	02	01	0C
		error-status	10	02	01	0A(wrongType)
		error-index	1	02	01	01
				30	*	
		variable-bindings		30	0D	
			1.3.6.1.2.1.1.5.0 sysName.0	06	08	2b 06 01 02 01 01 05 00
			100	02	01	64
				30	*	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	*	02	*	*

5th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	34	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00

			value	NULL	05	00	

6th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address			TN_ADDRESS			
	Destination Address			NUT_ADDRESS			
UDP Header	Source Port			161			
	Destination Port			Same as the source port in 1st packet			
SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	39	
	version			SNMPv2C	02	01	01
	community			private	04	07	70 72 69 76 61 74 65
	Data	PDU type		Response	A2	2C	
		request-id		12	02	01	0C
		error-status		0	02	01	0
		error-index		0	02	01	0
					30	21	
		variable-bindings			30	10	
				1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
				*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	*	02	01	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest scalar correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status and error-index is 0
4. two instances must be in the variable binding list with their correct syntax types and values within their defined range

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

5. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
6. request-id is the same as the request id in the previously received SNMPv2C SetRequest
7. error-status is notWritable and error-index is 1

OP3: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with

8. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
9. request-id is the same as the request id in the previously received SNMPv2C SetRequest
10. error-status and error-index are equal to zero
11. variable-bindings list must match the values received in OP1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

v6SNMPv2C4.5.7 Set read-only and read-write variables

Purpose

Verify that NUT playing the SNMPv2C agent can properly handle SetRequest PDU setting on hybrid read-only and read-write variables from SNMPv2C manager and return Response PDU correctly.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

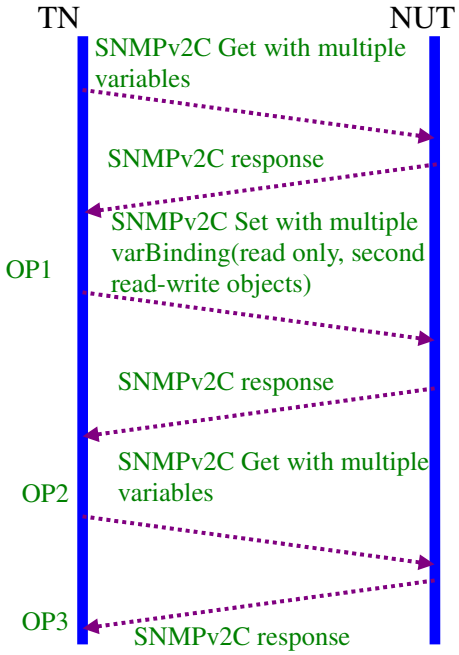
Please refer Fig 5. Test Architecture.

Setup

Refer Fig. 3 Common IPv6 Link Test Setup Basic before SNMPv2C Testing

Procedure

The test sequence is as follows



1. TN sends SNMPv2C GetRequest with multiple variables to NUT to save pre SetRequest OID value which will be restored afterwards.
2. NUT replies SNMPv2C Response before the SetRequest
3. TN sends SNMPv2C SetRequest with multiple variables to NUT.
4. NUT replies SNMPv2C Response with error code
5. TN sends SNMPv2C Get Request with multiple variables to NUT to check if the variables are correctly set by the Set operation
6. NUT replies SNMPv2C Response with current values

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	35	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
		name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
		value	NULL	05	00	
				30	0C	
		name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
		value	NULL	05	00	

2nd Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			161		
	Destination Port			Same as the source port in 1st packet		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	Response	A2	*	
		request-id	12	02	01	0C
		error-status	0	02	01	0
		error-index	0	02	01	0

					30	*	
		variable-bindings			30	*	
				1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
				*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	*	04	01	*

3rd Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	*	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	*	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	snmp_agent	04	0A	73 6e 6d 70 5F 61 67 65 6E 74
					30	0D	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	***test***	***04***	***04***	***74 65 73 74***

4th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)			
IP Header	Source Address		TN_ADDRESS
	Destination Address		NUT_ADDRESS
UDP Header	Source Port		any
	Destination Port		161
SNMP	SNMP Fields	Values	ASN.1(Hex)

Message			(readable)	type	len	value
				30	*	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	SetRequest	A3	*	
		request-id	12	02	01	0C
		error-status	17	02	01	11(notWritable)
		error-index	***1***	***02***	***01***	***01***
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	snmp_agent	04	0A	73 6e 6d 70 5F 61 67 65 6E 74
					30	0D	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	***test***	***04***	***04***	***74 65 73 74***

5th Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)						
IP Header	Source Address			TN_ADDRESS		
	Destination Address			NUT_ADDRESS		
UDP Header	Source Port			any		
	Destination Port			161		
SNMP Message	SNMP Fields		Values (readable)	ASN.1(Hex)		
				type	len	value
				30	35	
	version		SNMPv2C	02	01	01
	community		private	04	07	70 72 69 76 61 74 65
	Data	PDU type	GetRequest	A1	27	
		request-id	12	02	01	0C
		error-status	0	02	01	00
		error-index	0	02	01	00
				30	1C	
		variable-bindings		30	0C	
			name	1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
			value	NULL	05	00	
					30	0C	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	NULL	05	00	

6th Packet

Standard query response from SNMP agent (NUT) to SNMP manager (TN)							
IP Header	Source Address				TN_ADDRESS		
	Destination Address				NUT_ADDRESS		
UDP Header	Source Port				161		
	Destination Port				Same as the source port in 1st packet		
SNMP Message	SNMP Fields			Values (readable)	ASN.1(Hex)		
					type	len	value
					30	*	
	version			SNMPv2C	02	01	01
	community			private	04	07	70 72 69 76 61 74 65
	Data	PDU type		Response	A2	*	
		request-id		12	02	01	0C
		error-status		0	02	01	0
		error-index		0	02	01	0
					30	*	
		variable-bindings			30	*	
				1.3.6.1.2.1.1.1.0 (sysDescr.0)	06	08	2b 06 01 02 01 01 01 00
				*	04	*	*
					30	*	
			name	1.3.6.1.2.1.1.5.0 (sysName.0)	06	08	2b 06 01 02 01 01 05 00
			value	*	04	01	*

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest scalar correctly.

Received packet with

1. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
2. request-id is the same as the request id in the previously received SNMPv2C SetRequest
3. error-status and error-index must be equal to zero
4. two instances must be in the variable binding list with their correct syntax types and values within their defined range

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C SetRequest correctly.

Received packet with

5. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
6. request-id is the same as the request id in the previously received SNMPv2C SetRequest
7. error-status is notWritable and error-index is 1

OP3: TN received SNMPv2C response from NUT responding to SNMPv2C GetRequest correctly.

Received packet with

8. SNMP version = 1, Community=same as NUT's SNMPv2C community, PDU type =A2
9. request-id is the same as the request id in the previously received SNMPv2C SetRequest
10. error-status and error-index are equal to zero
11. variable-bindings list must match the values received in OP1

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.5

Group 5 IPv6 SNMPv2C Trap

Scope
The following tests verify the Trap command in IPv6 SNMPv2C protocols.

Overview
The SNMPv2-Trap-PDU is initiated by a SNMPv2C agent to generate generic traps. SNMPv2C manager, upon receiving such Trap-PDU, should correctly parse these trap types and act accordingly. No acknowledge is expected from the SNMPv2C manager for this trap operation. In this test, SNMPv2C agent must at least generate cold start and linkUp/LinkDown conditions and send a SNMPv2C trap PDU to the manager.

v6SNMPv2C5.1 Trap Test

Purpose

Verify that NUT playing as a SNMPv2C agent can properly generate the SNMPv2-Trap PDU and send it to the manager.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

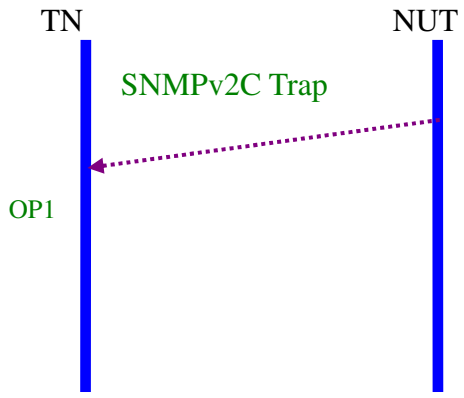
Please refer Fig 5. Test Architecture.

Setup

1. Refer Common Test Setup
2. The NUT operator must generate the power off situation for NUT if necessary

Procedure

The SNMPv2C Trap is an operation issued from SNMPv2C agent to SNMPv2C manager, and no confirmation from the manager is expected.



1. NUT sends SNMPv2C Trap PDU to TN.
2. No acknowledgement from the manager is expected.

1st Packet

Standard query from SNMP manager (TN) to SNMP agent (NUT)					
IP Header	Source Address		NUT_ADDRESS		
	Destination Address		TN_ADDRESS		
UDP Header	Source Port		any		
	Destination Port		162		
SNMP Message	SNMP Fields	Values (readable)	ASN.1(Hex)		
			type	len	value
			30	*	
	version	SNMPv2C	02	01	01

	community		public	04	06	70 75 62 6C 69 63
	Data	PDU type	Trap	A7	*	
		request-id		02	01	*
		error-status		02	01	00
		error-index		02	01	00
				30	*	
		variable-bindings		30	*	
			name	1.3.6.1.2.1.1.3.0 (sysUpTime.0)	06	08	2b 06 01 02 01 01 03 00
			value	TimeTicks	02	*	TimeTicks*
					30	*	
			name	1.3.6.1.6.3.1.1.4.1.0 (snmpTrapOID.0)	06	0A	2b 06 01 06 03 01 01 04 01 00
			value	Trap OID	06	*	*

Note * Trap OID can be either coldStart, warmStart, linkDown, linkUp as defined in snmpTrapOID

Exp.
NUT_Address: SNMPv2C agent (NUT) address
TN_Address: SNMPv2C manager (TN) address

Judgment

OP1: TN received the SNMPv2C Trap with sysUpTime, sysTrapOID. The latter defined the cause reason either with coldStart, warmStart, linkDown or linkup.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2.6

Section 2 Management Information Base

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP).

General Test and Judgment Rules

A SNMP walk should be performed to query for a subtree of information about a node. All variables in the subtree below the given mib-group-name are queried and their values will be presented.

The returned variable binding name and value for each tested MIB OID must have correct ASN.1 coding. Check for the .0 for each returned scalar OID. For tabular objects, the table index must be correct. All the returned MIB variable values must have the correct syntax type and within the defined range value.

Only those mandatory test items are marked as B(basic) in the following MIB table are judged for passing the test or not. A stands for Advanced(optional) test items.

RFC 3418 SNMPv2 MIB

Scope

This test is to test managed objects which describe the behavior of an SNMP entity, as defined in RFC 3418.

Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). The tested MIB in this document is compliant to the SMIv2 described in RFC 2578, RFC 2579 and RFC 2580.

Overview

These tests are designed to verify the readiness of a SNMPv2 MIB implementation.

Group 1 verify the implementation of object identifiers

v6SNMPv2CMIB1.1 System Group

Purpose

This test shall verify that NUT has implemented general objects correctly. Only basic(mandatory) objects are used for judgment criteria. Table 2 is the MIB II System Group Test Criteria. B stands for basic test items and A stands for advanced test items

Table 2 MIB II System Group Test Criteria

Name	OID	MAX-Access	Syntax	Required
sysDescr	1.3.6.1.2.1.1.1	RO	Octet string	B
sysObjectID	1.3.6.1.2.1.1.2	RO	OBJECT IDENTIFIER	B
sysUpTime	1.3.6.1.2.1.1.3	RO	TimeTicks	B
sysContact	1.3.6.1.2.1.1.4	RW	Octets	B
sysName	1.3.6.1.2.1.1.5	RW	Octet string	B
sysLocation	1.3.6.1.2.1.1.6	RW	Octets	B
sysServices	1.3.6.1.2.1.1.7	RO	Integer(32Bits)	B
sysORLastChange	1.3.6.1.2.1.1.8	RO	TimeStamp	A
sysORTable	1.3.6.1.2.1.1.9	NA	SEQUENCE OF sysOREntry	A
sysOREntry	1.3.6.1.2.1.1.9.1	NA	sysOREntry	A
sysORIndex	1.3.6.1.2.1.1.9.1.1	NA	INTEGER	A
sysORID	1.3.6.1.2.1.1.9.1.2	RO	OBJECT IDENTIFIER	A
sysORDescr	1.3.6.1.2.1.1.9.1.3	RO	Octets sysORUpTime	A
sysORUpTime	1.3.6.1.2.1.1.9.1.4	RO	TimeTicks	A

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

NUT shall perform a general test as described in General Test and Judgment rules on this system group.

Judgment

Value field is the OID with correct syntax type and value within the defined range field.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 3418, Management Information Base (MIB) for the Simple Network Management Protocol (SNMP), Page 3-6

v6SNMPv2CMIB1.2 SNMP Group

Purpose

This test shall verify that NUT has implemented mandatory objects correctly using the SNMPv2C GetRequest command. Only B (mandatory) objects are used for judgment criteria. Table 3 is the test criteria for MIB II SNMP Group.

Table 3 MIB II SNMP Group Test Criteria

Name	OID	MAX-Access	Syntax	Required
snmpInPkts	1.3.6.1.2.1.11.1	RO	Counter32	A
snmpInBadVersions	1.3.6.1.2.1.11.3	RO	Counter32	A
snmpInBadCommunityNames	1.3.6.1.2.1.11.4	RO	Counter32	A
snmpInBadCommunityUses	1.3.6.1.2.1.11.5	RO	Counter32	A
snmpInASNParseErrs	1.3.6.1.2.1.11.6	RO	Counter32	A
snmpEnableAuthenTraps	1.3.6.1.2.1.11.30	RW	INTEGER {enabled(1), disabled(2)}	A
snmpSilentDrops	1.3.6.1.2.1.11.31	RO	Counter32	A
snmpProxyDrops	1.3.6.1.2.1.11.32	RO	Counter32	A

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN performs a general test on NUT for the snmp group and examines each object identifier for the correct syntax and valid range check.

Judgment

Value field is the OID with correct syntax type and value within the defined range field.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol

RFC 3418, Management Information Base (MIB) for the Simple Network Management Protocol (SNMP), Page 7-9

v6SNMPv2CMIB1.2.1 SNMPInPkts counter check

Purpose

This test shall verify that NUT has implemented snmpInPkts(1.3.6.1.2.1.11.1) correctly using the SNMPv2C GetRequest command.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

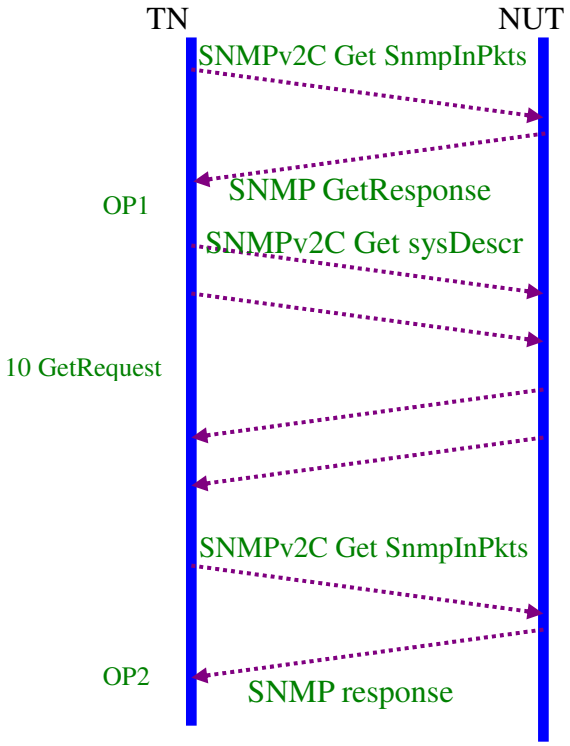
Setup

Refer Common Test Setup

Procedure

The test sequence is as follows

1. TN sends SNMPv2C GetRequest scalar object to NUT by issuing SNMPv2C Get to get snmpInPkts(1.3.6.1.2.1.1.1.1.0)
2. NUT replies SNMPv2C Response with correct variable binding pairs to TN
3. TN sends ten SNMPv2C GetRequest with sysDescr.0(1.3.6.1.2.1.1.1.0)
4. TN send SNMPv2C GetRequest snmpInPkts (1.3.6.1.2.1.11.1) again to check its value



Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C Get object request with snmpInPkts(snmpInPkts1) before the 10 GetRequest sending

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C Get object request with incremented snmpInPkts(snmpInPkts2) value after the 10 GetRequest sending, i.e. snmpInPkts2=snmpInPkts1+11

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol

RFC 3418, Management Information Base (MIB) for the Simple Network Management Protocol (SNMP), Page 7-9

v6SNMPv2CMIB1.2.2 snmpSilentDrops counter check

Purpose

Verify that NUT playing SNMPv2C agent can properly detect the SNMPv2C GetRequest with invalid sequence_of type field in the received packet from the SNMPv2C manager and increment the snmpSilentDrops(1.3.6.1.2.1.11.31 as defined in RFC3418) counter after it discards the datagram.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

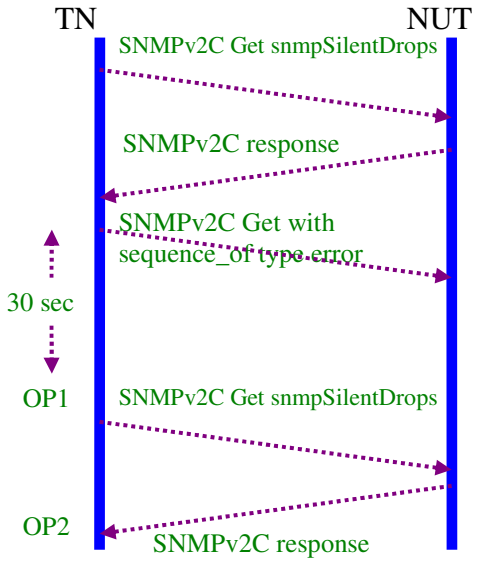
Setup

Refer Common Test Setup

Procedure

The test sequence is as follows:

1. TN sends SNMPv2C GetRequest snmpSilentDrops
2. NUT return the snmpSilentDrops value before the error test.
3. TN sends SNMPv2C GetRequest with sequence_of type error to NUT.
4. NUT discards the datagram and continues to respond to normal requests.
5. TN sends SNMPv2C GetRequest snmpSilentDrops to NUT to check if counter has been incremented by one.
6. NUT returns the latest snmpSilentDrops value.



Judgment

OP1: NUT will silently discard this malformed SNMPv2C GetRequest with sequence_of type error packet.

OP2: snmpSilentDrops counter is correctly incremented by one, i.e.

snmpSilentDropsCounter2 = snmpSilentDropsCounter1 + 1.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol, Sec 4.2

RFC 3418, Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)

RFC 4293 IP-MIB

Scope

The following conformance tests follow and cover RFC 4293-Management Information Base for Internet Protocol specification.

Overview

These tests are designed to verify the readiness of a new RFC 4293 MIB implementation.

Group 2 verify the implementation of object identifiers

Scope

The following tests verify the implementation of object identifiers in RFC 4293 that are IPv6 related. Only those that are marked as B(basic, mandatory) in the following MIB table are checked for passing the test or not.

Overview

Tests in this group verify that the implementation of object identifiers in RFC 4293 that are IPv6 related is correct. These IPv6 related object identifiers are included in the IP interfaces table, the IP statistics table, the internet address prefix table, the internet address table, and IPv6 Scope Zone Index Table the default router table, Router Advertisement Table and ICMP Statistics Tables. Only B(ASIC for mandatory) objects are selected for testing judgment. Selection of tested OIDs is based on RFC 4293 Conformance and Compliance. All mandatory groups are mandatory. Only IPv6 related OIDs are tested.

v6SNMPv2CMIB2.1 General Objects

Purpose

This test shall verify that NUT has implemented general objects correctly using the SNMPv2C GetRequest command. Only IPv6 related are tested. B stands for mandatory and these objects shall be used for judgment criteria.

Table 4 is the test criteria for this RFC 4293 IP MIB General Group test

Table 4 RFC 4293 IP MIB – General Group Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ip general group					
ipForwarding	1.3.6.1.2.1.4.1	RW	INTEGER {forwarding(1), notForwarding(2) }	-	-
ipDefaultTTL	1.3.6.1.2.1.4.2	RW	Integer32(1..255)	-	-
ipReasmTimeout	1.3.6.1.2.1.4.13	RO	Integer32	-	-
ipv6 general group					
ipv6IpForwarding	1.3.6.1.2.1.4.25	RW	INTEGER {forwarding(1), notForwarding(2) }	B	B
ipv6IpDefaultHopLimit	1.3.6.1.2.1.4.26	RW	Integer32(0..255)	B	B

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN shall get NUT's ipv6IpForwarding(1.3.6.1.2.1.4.25) and ipv6IpDefaultHopLimit (1.3.6.1.2.1.4.26) values.

Judgment

Examine the return OID values for each basic(mandatory) object identifier for valid syntax type and value range.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.1

v6SNMPv2CMIB2.2 InterfaceTables

Purpose

This test shall verify that NUT has implemented interface tables correctly. Table 5 is the test criteria for RFC 4293 IP MIB – InterfaceTable test. Only B(mandatory) objects are used for judgment criteria.

Table 5 RFC 4293 IP MIB – InterfaceTable Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ipv4InterfaceTableLastChange	1.3.6.1.2.1.4.27	RO	TimeStamp	-	-
ipv4InterfaceTable	1.3.6.1.2.1.4.28	NA	SWQUENCE OF Ipv4InterfaceEntry	-	-
ipv4InterfaceEntry	1.3.6.1.2.1.4.28.1	NA	Ipv4InterfaceEntry	-	-
ipv4InterfaceIfIndex	1.3.6.1.2.1.4.28.1.1	NA	InterfaceIndex	-	-
ipv4InterfaceReasmMaxSize	1.3.6.1.2.1.4.28.1.2	RO	Unsigned32 (0..65535)	-	-
ipv4InterfaceEnableStatus	1.3.6.1.2.1.4.28.1.3	RW	INTEGER(up(1), down(2))	-	-
ipv4InterfaceRetransmitTime	1.3.6.1.2.1.4.28.1.4	RO	Unsigned32 Defval=1000	-	-
ipv6InterfaceTableLastChange	1.3.6.1.2.1.4.29	RO	TimeStamp	A	A
ipv6InterfaceTable	1.3.6.1.2.1.4..30	NA	SEQUENCE OF Ipv6InterfaceEntry	B	B
ipv6InterfaceEntry	1.3.6.1.2.1.4.30.1	NA	Ipv6InterfaceEntry	B	B
ipv6InterfaceIfIndex	1.3.6.1.2.1.4.30.1.1	NA	InterfaceIfIndex	B	B
ipv6InterfaceReasmMaxSize	1.3.6.1.2.1.4.30.1.2	RO	Unsigned32 (1500..65535)	B	B
ipv6InterfaceIdentifier	1.3.6.1.2.1.4.30.1.3	RO	Ipv6AddressIfIdentifierTC	B	B
ipv6InterfaceEnableStatus	1.3.6.1.2.1.4.30.1.5	RW	Integer(1: up(1), 2: down(2))	B	B
ipv6InterfaceReachableTime	1.3.6.1.2.1.4.30.1.6	RO	Unsigned32	B	B
ipv6InterfaceRetransmitTime	1.3.6.1.2.1.4.30.1.7	RO	Unsigned32	B	B
ipv6InterfaceForwarding	1.3.6.1.2.1.4.30.1.8	RW	INTEGER {forwarding(1), notForwarding(2) }	B	B

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN shall perform a GetNext walk on NUT's Interface Table.

Judgment

Examine the return OID values for each B (mandatory) object identifier for valid syntax type and value range.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol

RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.2

v6SNMPv2CMIB2.3 IP Statistics Tables

Purpose

This test shall verify that NUT has implemented IP statistics tables (ipSystemStatsTable and ipIfStatsTable) which contain objects to count the number of datagrams and octets that a given entity has processed. Table 6 is the test criteria for this RFC 4293 IP MIB – IP traffic statistics Table test. ipSystemStatsTable is mandatory and ipIfStatsTable is optional in this test. B(mandatory) objects in Table 6, shall be used for judgment criteria.

Table 6 RFC 4293 IP MIB – IP traffic statistics Table Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ipSystemStatsTable	1.3.6.1.2.1.4.31.1	NA	SEQUENCE OF IpSystemStatsEntry	B	B
ipSystemStatsEntry	1.3.6.1.2.1.4.31.1.1	NA	IpSystemStatsEntry	B	B
ipSystemStatsIPversion	1.3.6.1.2.1.4.31.1.1.1	NA	InetVersion {ipv4(1), ipv6(2)}	B	B
ipSystemStatsInReceives	1.3.6.1.2.1.4.31.1.1.3	RO	Counter32	B	B
ipSystemStatsHCInReceives	1.3.6.1.2.1.4.31.1.1.4	RO	Counter64	B	B
ipSystemStatsInOctets	1.3.6.1.2.1.4.31.1.1.5	RO	Counter32	B	B
ipSystemStatsHCInOctets	1.3.6.1.2.1.4.31.1.1.6	RO	Counter64	B	B
ipSystemStatsInHdrErrors	1.3.6.1.2.1.4.31.1.1.7	RO	Counter32	B	B
ipSystemStatsInNoRoutes	1.3.6.1.2.1.4.31.1.1.8	RO	Counter32	B	B
ipSystemStatsInAddrErrors	1.3.6.1.2.1.4.31.1.1.9	RO	Counter32	B	B
ipSystemStatsInUnknownProtos	1.3.6.1.2.1.4.31.1.1.10	RO	Counter32	B	B
ipSystemStatsInTruncatedPkts	1.3.6.1.2.1.4.31.1.1.11	RO	Counter32	B	B
ipSystemStatsInForwDatagrams	1.3.6.1.2.1.4.31.1.1.12	RO	Counter32	B	B
ipSystemStatsHCInForwDatagrams	1.3.6.1.2.1.4.31.1.1.13	RO	Counter64	B	B
ipSystemStatsReasmReqds	1.3.6.1.2.1.4.31.1.1.14+	RO	Counter32	B	B
ipSystemStatsReasmOKs	1.3.6.1.2.1.4.31.1.1.15	RO	Counter32	B	B
ipSystemStatsReasmFails	1.3.6.1.2.1.4.31.1.1.16	RO	Counter32	B	B
ipSystemStatsInDiscards	1.3.6.1.2.1.4.31.1.1.17	RO	Counter32	B	B
ipSystemStatsInDelivers	1.3.6.1.2.1.4.31.1.1.18	RO	Counter32	B	B
ipSystemStatsHCInDelivers	1.3.6.1.2.1.4.31.1.1.19	RO	Counter64	B	B

ipSystemStatsOutRequests	1.3.6.1.2.1.4.31.1.1.20	RO	Counter32	B	B
ipSystemStatsHCOutRequests	1.3.6.1.2.1.4.31.1.1.21	RO	Counter64	B	B
ipSystemStatsOutNoRoutes	1.3.6.1.2.1.4.31.1.1.22	RO	Counter32	B	B
ipSystemStatsOutForwDatagrams	1.3.6.1.2.1.4.31.1.1.23	RO	Counter32	B	B
ipSystemStatsHCOutForwDatagrams	1.3.6.1.2.1.4.31.1.1.24	RO	Counter64	B	B
ipSystemStatsOutDiscards	1.3.6.1.2.1.4.31.1.1.25	RO	Counter32	B	B
ipSystemStatsOutFragReqds	1.3.6.1.2.1.4.31.1.1.26	RO	Counter32	B	B
ipSystemStatsOutFragOKss	1.3.6.1.2.1.4.31.1.1.27-	RO	Counter32	B	B
ipSystemStatsOutFragFails	1.3.6.1.2.1.4.31.1.1.28	RO	Counter32	B	B
ipSystemStatsOutFragCreates	1.3.6.1.2.1.4.31.1.1.29	RO	Counter32	B	B
ipSystemStatsOutTransmits	1.3.6.1.2.1.4.31.1.1.30	RO	Counter32	B	B
ipSystemStatsHCOutTransmits	1.3.6.1.2.1.4.31.1.1.31	RO	Counter64	B	B
ipSystemStatsOutOctets	1.3.6.1.2.1.4.31.1.1.32	RO	Counter32	B	B
ipSystemStatsHCOutOctets	1.3.6.1.2.1.4.31.1.1.33	RO	Counter64	B	B
ipSystemStatsInMcastPkts	1.3.6.1.2.1.4.31.1.1.34	RO	Counter32	B	B
ipSystemStatsHCInMcastPkts	1.3.6.1.2.1.4.31.1.1.35	RO	Counter64	B	B
ipSystemStatsInMcastOctets	1.3.6.1.2.1.4.31.1.1.36	RO	Counter32	B	B
ipSystemStatsHCInMcastOctets	1.3.6.1.2.1.4.31.1.1.37	RO	Counter64	B	B
ipSystemStatsOutMcastPkts	1.3.6.1.2.1.4.31.1.1.38	RO	Counter32	B	B
ipSystemStatsHCOutMcastPkts	1.3.6.1.2.1.4.31.1.1.39	RO	Counter64	B	B
ipSystemStatsOutMcastOctets	1.3.6.1.2.1.4.31.1.1.40	RO	Counter32	B	B
ipSystemStatsHCOutMcastOctets	1.3.6.1.2.1.4.31.1.1.41	RO	Counter64	B	B
ipSystemStatsInBcastPkts	1.3.6.1.2.1.4.31.1.1.42	RO	Counter32	B	B
ipSystemStatsHCInBcastPkts	1.3.6.1.2.1.4.31.1.1.43	RO	Counter64	B	B
ipSystemStatsOutBcastPkts	1.3.6.1.2.1.4.31.1.1.44	RO	Counter32	B	B
ipSystemStatsHCOutBcast	1.3.6.1.2.1.4.31.1.1.45	RO	Counter64	B	B

Pkts					
ipSystemStatsDiscontinuityTime	1.3.6.1.2.1.4.31.1.1.46	RO	Counter32	B	B
ipSystemStatsRefreshRate	1.3.6.1.2.1.4.31.1.1.47	RO	Counter32	B	B
ipIfStatsTableLastChange	1.3.6.1.2.1.4.31.2	RO	TimeStamp	A	A
ipIfStatsTable	1.3.6.1.2.1.4.31.3	NA	SEQUENCE OF ipIfStatsEntry	A	A
ipIfStatsEntry	1.3.6.1.2.1.4.31.3.1	NA	ipIfStatsEntry	A	A
ipIfStatsIPversion	1.3.6.1.2.1.4.31.3.1.1	NA	InetVersion {ipv4(1), ipv6(2)}	A	A
ipIfStatsIfIndex	1.3.6.1.2.1.4.31.3.1.2	NA	InterfaceIndex	A	A
ipIfStatsInReceives	1.3.6.1.2.1.4.31.3.1.3	RO	Counter32	A	A
ipIfStatsHCInReceives	1.3.6.1.2.1.4.31.3.1.4	RO	Counter64	A	A
ipIfStatsInOctets	1.3.6.1.2.1.4.31.3.1.5	RO	Counter32	A	A
ipIfStatsHCInOctets	1.3.6.1.2.1.4.31.3.1.6	RO	Counter64	A	A
ipIfStatsInHdrErrors	1.3.6.1.2.1.4.31.3.1.7	RO	Counter32	A	A
ipIfStatsInNoRoutes	1.3.6.1.2.1.4.31.3.1.8	RO	Counter32	A	A
ipIfStatsInAddrErrors	1.3.6.1.2.1.4.31.3.1.9	RO	Counter32	A	A
ipIfStatsInUnknownProtos	1.3.6.1.2.1.4.31.3.1.10	RO	Counter32	A	A
ipIfStatsInTruncatedPkts	1.3.6.1.2.1.4.31.3.1.11	RO	Counter32	A	A
ipIfStatsInForwDatagrams	1.3.6.1.2.1.4.31.3.1.12	RO	Counter32	A	A
ipIfStatsHCInForwDatagrams	1.3.6.1.2.1.4.31.3.1.13	RO	Counter32	A	A
ipIfStatsReasmReqds	1.3.6.1.2.1.4.31.3.1.14	RO	Counter32	A	A
ipIfStatsReasmOKs	1.3.6.1.2.1.4.31.3.1.15	RO	Counter32	A	A
ipIfStatsReasmFails	1.3.6.1.2.1.4.31.3.1.16	RO	Counter32	A	A
ipIfStatsInDiscards	1.3.6.1.2.1.4.31.3.1.17	RO	Counter32	A	A
ipIfStatsInDelivers	1.3.6.1.2.1.4.31.3.1.18	RO	Counter32	A	A
ipIfStatsHCInDelivers	1.3.6.1.2.1.4.31.3.1.19	RO	Counter64	A	A
ipIfStatsOutRequests	1.3.6.1.2.1.4.31.3.1.20	RO	Counter32	A	A
ipIfStatsHCOutRequests	1.3.6.1.2.1.4.31.3.1.21	RO	Counter64	A	A
ipIfStatsOutForwDatagrams	1.3.6.1.2.1.4.31.3.1.23	RO	Counter32	A	A
ipIfStatsHCOutForwDatagrams	1.3.6.1.2.1.4.31.3.1.24	RO	Counter64	A	A
ipIfStatsOutDiscards	1.3.6.1.2.1.4.31.3.1.25	RO	Counter32	A	A
ipIfStatsOutFragReqds	1.3.6.1.2.1.4.31.3.1.26	RO	Counter32	A	A
ipIfStatsOutFragOKs	1.3.6.1.2.1.4.31.3.1.27	RO	Counter32	A	A
ipIfStatsOutFragFails	1.3.6.1.2.1.4.31.3.1.28	RO	Counter32	A	A
ipIfStatsOutFragCreates	1.3.6.1.2.1.4.31.3.1.29	RO	Counter32	A	A
ipIfStatsOutTransmits	1.3.6.1.2.1.4.31.3.1.30	RO	Counter32	A	A

ipIfStatsHCOutTransmits	1.3.6.1.2.1.4.31.3.1.31	RO	Counter64	A	A
ipIfStatsOutOctets	1.3.6.1.2.1.4.31.3.1.32	RO	Counter32	A	A
ipIfStatsHCOutOctets	1.3.6.1.2.1.4.31.3.1.33	RO	Counter64	A	A
ipIfStatsInMcastPkts	1.3.6.1.2.1.4.31.3.1.34	RO	Counter32	A	A
ipIfStatsHCInMcastPkts	1.3.6.1.2.1.4.31.3.1.35	RO	Counter64	A	A
ipIfStatsInMcastOctets	1.3.6.1.2.1.4.31.3.1.36	RO	Counter32	A	A
ipIfStatsHCInMcastOctets	1.3.6.1.2.1.4.31.3.1.37	RO	Counter64	A	A
ipIfStatsOutMcastPkts	1.3.6.1.2.1.4.31.3.1.38	RO	Counter32	A	A
ipIfStatsHCOutMcastPkts	1.3.6.1.2.1.4.31.3.1.39	RO	Counter64	A	A
ipIfStatsOutMcastOctets	1.3.6.1.2.1.4.31.3.1.40	RO	Counter32	A	A
ipIfStatsHCOutMcastOctets	1.3.6.1.2.1.4.31.3.1.41	RO	Counter64	A	A
ipIfStatsInBcastPkts	1.3.6.1.2.1.4.31.3.1.42	RO	Counter32	A	A
ipIfStatsHCInBcastPkts	1.3.6.1.2.1.4.31.3.1.43	RO	Counter64	A	A
ipIfStatsOutBcastPkts	1.3.6.1.2.1.4.31.3.1.44	RO	Counter32	A	A
ipIfStatsHCOutBcastPkts	1.3.6.1.2.1.4.31.3.1.45	RO	Counter64	A	A
ipIfStatsDiscontinuityTime	1.3.6.1.2.1.4.31.3.1.46	RO	Counter32	A	A
ipIfStatsRefreshRate	1.3.6.1.2.1.4.31.3.1.47	RO	Counter32	A	A

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

The test sequence is as follows

1. NUT performs a GetNext walk on IP traffic statistics Tables
2. Examine each object identifier for the correct syntax and valid range check.

Judgment

The return OID values for each B (mandatory) object identifier are with valid syntax type and within defined value range.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.3

v6SNMPv2CMIB2.3.1 ipSystemStatsInOctes counter check

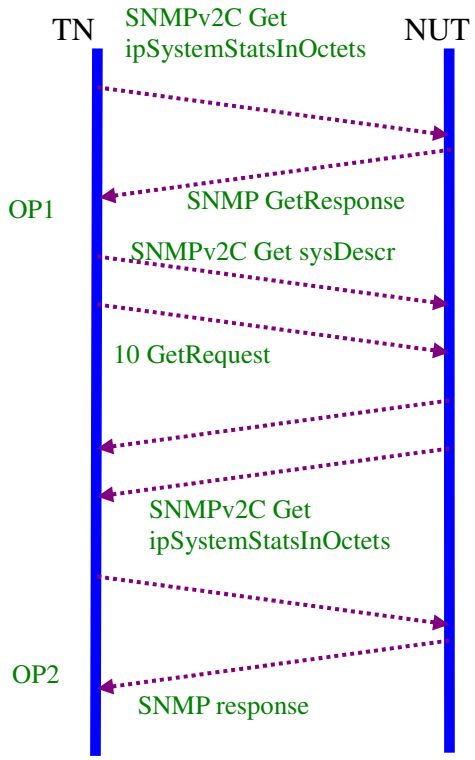
Purpose

ipSystemStatsInOctets in Table 6 shall be checked. Ten GetRequests will be sent to see if the ipSystemStatsInOctets is correctly incremented.

Procedure

The test sequence is as follows

1. TN sends SNMPv2C Get scalar object to NUT by issuing SNMPv2C Get to get ipSystemStatsInOctets(1.3.6.1.2.1.4.31.1.1.5.2).
2. NUT replies SNMPv2C Response with correct variable binding pairs to TN
3. TN send ten SNMPv2C Get scalar object to NUT by issuing SNMPv2C GetRequest with sysDescr(1.3.6.1.2.1.1.1.1.0)
4. TN sends SNMPv2C Get ipSystemStatsInOctets again to check its value



Judgment

OP1: TN received SNMPv2C response from NUT responding to SNMPv2C Get object request with ipSystemStatsInOctets(ipSystemStatsInOctets1) value before the 10 GetRequest sending

OP2: TN received SNMPv2C response from NUT responding to SNMPv2C Get object request with incremented ipSystemStatsInOctets(ipSystemStatsInOctets2) value after the 10 GetRequest sending i.e. ipSystemStatsInOctets2= ipSystemStatsInOctets1+10*(40+8+40) + 1*(40+8+43).

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.3

v6SNMPv2CMIB2.4 Internet Address Prefix Table

Purpose

Internet Address Prefix Table provides information about the prefixes this entity is using, including their lifetimes. This table provides a convenient place to which other tables that make use of prefixes, such as the ipAddressTable, may point. By including this table, the MIB can supply the prefix information for all addresses, yet minimize the amount of duplication required in storing and accessing this data. This arrangement also clarifies the relationship between addresses that have the same prefix. This table is required for IPv6 entities. Table 7 is RFC 4293 IP MIB – IP address Prefix Table Test Criteria.

This test shall existence of Internet Address Prefix Table and verifies the OID values in this table. B (mandatory) objects are used for judgment criteria.

Table 7 RFC 4293 IP MIB – IP Address Prefix Table Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ipAddressPrefixTable	1.3.6.1.2.1.4.32.	NA	SEQUENCE OF IpAddressPrefixEntry	B	B
ipAddressPrefixEntry	1.3.6.1.2.1.4.32.1.	NA	IpAddressPrefixEntry	B	B
ipAddressPrefixIfIndex	1.3.6.1.2.1.4.32.1.1	NA	InterfaceIndex	B	B
ipAddressPrefixType	1.3.6.1.2.1.4.32.1.2	NA	InetAddressType {ipv4{1}, ipv6(2)}	B	B
ipAddressPrefixPrefix	1.3.6.1.2.1.4.32.1.3	NA	InetAddress (Size(4|16))	B	B
ipAddressPrefixLength	1.3.6.1.2.1.4.32.1.4	NA	Unsigned32	B	B
ipAddressPrefixOrigin	1.3.6.1.2.1.4.32.1.5	RO	INTEGER {other(1), manual (2), wellknown (3), dhcp (4), routeradv (5)}	B	B
ipAddressPrefixOnLinkFlag	1.3.6.1.2.1.4.32.1.6	RO	Truthvalue (default=True(1))	B	B
ipAddressPrefixAutonomousFlag	1.3.6.1.2.1.4.32.1.7	RO	Truthvalue	B	B
ipAddressPrefixAdvPreferredLifetime	1.3.6.1.2.1.4.32.1.8	RO	Unsigned32	B	B
ipAddressPrefixAdvvalidLifetime	1.3.6.1.2.1.4.32.1.9	RO	Unsigned32	B	B

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN performs a GetNext walk for NUT's IP Address Prefix Table.

Judgment

Examine the return OID values for each B(mandatory) object identifier for valid syntax type and value range.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.4

v6SNMPv2CMIB2.5 Internet Address Table

Purpose

The Internet address Table lists the IP addresses (both IPv4 and IPv6) used by this entity. It also includes some basic information about how and when the address was formed and last updated. This table is required for all IP entities. Table 8 is the test criteria for conducting this RFC 4293 IP MIB Internet Address Table MIB test.

This test shall existence of Internet Address Table and verifies the OID values in this table. B (mandatory) objects shall be used for judgment criteria.

Table 8 RFC 4293 IP MIB – Internet Address Table Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ipAddressSpinLock	1.3.6.1.2.1.4.33	NA	TestAndIncr	B	B
ipAddressEntry	1.3.6.1.2.1.4.34.1	NA	ipAddressEntry	B	B
ipAddressAddrType	1.3.6.1.2.1.4.34.1.1	NA	InetAddressType {ipv4(1), ipv6(2)}	B	B
ipAddressAddr	1.3.6.1.2.1.4.34.1.2	NA	InetAddress (Size(4 | 8 | 16 | 20))	B	B
ipAddressIfIndex	1.3.6.1.2.1.4.34.1.3	RW	InterfaceIndex	B	B
ipAddressType	1.3.6.1.2.1.4.34.1.4	RC	INTEGER {unicast(1), anycast(2), broadcast(3)}	B	B
ipAddressPrefix	1.3.6.1.2.1.4.34.1.5	RO	RowPointer (Object Identifier)	B	B
ipAddressOrigin	1.3.6.1.2.1.4.34.1.6	RO	IpAddressOriginTC (INTEGER)	B	B
ipAddressStatus	1.3.6.1.2.1.4.34.1.7	RC	IpAddressStatusTC (INTEGER) {preferred(1)=default, deprecated(2), invalid(3), inaccessible(4), unknown(5), tentative(6), duplicate(7), optimistic(8) }	B	B
ipAddressCreated	1.3.6.1.2.1.4.34.1.8	RO	TimeStamp	B	B
ipAddressLastChanged	1.3.6.1.2.1.4.34.1.9	RO	TimeStamp	B	B
ipAddressRowStatus	1.3.6.1.2.1.4.34.1.10	RC	RowStatus	B	B
ipAddressStorageType	1.3.6.1.2.1.4.34.1.11	RC	StorageType(default=volatile, permanent)	B	B

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN performs a GetNext walk on NUT for IP Address Table.

Judgment

Examine the return OID values for each B(mandatory) object identifier for valid syntax type and value range

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.5

v6SNMPv2CMIB2.6.1 Internet Address Translation Table

Purpose

Internet Address Translation Table provides a mapping between IP layer addresses and physical addresses as would be formed by either Address Resolution Protocol (ARP) for IPv4 or the neighbor discovery protocol for IPv6.

This test shall existence of Internet Address Translation Table and verifies the OID values in this table. B stands for basic (mandatory) objects and they shall be used for judgment criteria. Table 9 is the test criteria list for conducting this RFC 4293 IP MIB –Address Translation Table test.

Table 9 RFC 4293 IP MIB –Address Translation Table Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ipNetToPhysicalTable	1.3.6.1.2.1.4.35	NA	SEQUENCE OF IpNetToPhysicalEntry	B	B
ipNetToPhysicalEntry	1.3.6.1.2.1.4.35.1	NA	IpNetToPhysicalEntry	B	B
ipNetToPhysicalIfIndex	1.3.6.1.2.1.4.35.1.1	NA	InterfaceIndex	B	B
ipNetToPhysicalNetAddressType	1.3.6.1.2.1.4.35.1.2	NA	InetAddressType {ipv4(1), ipv6(2), ipv4z(3), ipv6z(4)}	B	B
ipNetToPhysicalNetAddress	1.3.6.1.2.1.4.35.1.3	NA	InetAddress (Size(4 | 8 | 16 | 20))	B	B
ipNetToPhysicalPhysAddress	1.3.6.1.2.1.4.35.1.4	RC	PhysAddress (SIZE (0..65535))	B	B
ipNetToPhysicalLastUpdated	1.3.6.1.2.1.4.35.1.5	RO	TimeStamp	B	B
ipNetToPhysicalType	1.3.6.1.2.1.4.35.1.6	RC	INTEGER {other(1), invalid(2),dynamic(3), static(4), local(5)}	B	B
ipNetToPhysicalState	1.3.6.1.2.1.4.35.1.7	RO	INTEGER {reachable(1), stale(2), delay(3), probe(4), invalid(5), unknown(6),incomplete(7)}	B	B
ipNetToPhysicalRowStatus	1.3.6.1.2.1.4.35.1.8	RC	RowStatus	B	B

Resource Requirements

- Packet generator

- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN performs a general test on NUT for IP Address Translation Table.

Judgment

Examine the return OID values for each B(mandatory) object identifier for valid syntax type and value range.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.6

v6SNMPv2CMIB2.6.2 IPNetToPhysicalAddress Check

Purpose

IpNetToPhysicalAddress in Internet Address Translation Table will be checked to see if NUT can learn neighbor information correctly after TN, functioning as an emulated REF-NODE in the same LAN environment, performs ping operations on NUT.

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

1. Reboot NUT.
2. REF-NODE pings NUT with link-local address.
3. TN performs a SNMP-WALK on the ipNetToPhysicalPhyAddress to check
   - ✓ NUT does have the entry of ipNetToPhysicalPhyAddress.[link-local address] for REF-NODE.
   - ✓ NUT does not have the entry of ipNetToPhysicalPhyAddress.[global address] for REF-NODE.
4. REF-NODE pings NUT with global address.
5. REF-NODE waits for Echo Reply.
6. TN performs a SNMP-WALK on the ipNettoPHsicalPhyAddress to check NUT has the entry of ipNettoPhysicalPhyAddress.[global address] for REF-NODE.

Judgment

1. Test result from procedure 3 :
   - ✓ NUT does have the entry of ipNetToPhysicalPhyAddress.[link-local address] for REF-NODE.
   - ✓ NUT does not have the entry of ipNetToPhysicalPhyAddress.[global address] for REF-NODE.
2. Test result from procedure 5 :
   NUT has the entry of ipNetToPhysicalPhyAddress.[global address] for REF-NODE.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.6

v6SNMPv2CMIB2.7 IPv6 Scope Zone Index Table

Purpose

IPv6 Scope Zone Index Table specifies the zone index to interface mapping. By examining the table, a manager can determine which groups of interfaces are within a particular zone for a given scope. The zone index information is only valid within a given entity; the indexes used on one entity may not be comparable to those used on a different entity. This table is required for IPv6 entities.

This test shall check the existence of IPv6 Scope Zone Index Table and verify the OID values in this table. Table 10 lists the test criteria for conducting this IPv6 Scope Zone Index test. B (mandatory) objects shall be used for judgment criteria.

Table 10 RFC 4293 IP MIB – IPv6 Scope Zone Index Table Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ipv6ScopeZoneIndexTable	1.3.6.1.2.1.4.36	NA	SEQUENCE OF ipv6ScopeZoneIndexEntry	B	B
ipv6ScopeZoneIndexEntry	1.3.6.1.2.1.4.36.1	NA	ipv6ScopeZoneIndexEntry	B	B
ipv6ScopeZoneIndexIfIndex	1.3.6.1.2.1.4.36.1.1	NA	InterfaceIndex	B	B
ipv6ScopeZoneIndexLinkLocal	1.3.6.1.2.1.4.36.1.2	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndex3	1.3.6.1.2.1.4.36.1.3	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndexAdminLocal	1.3.6.1.2.1.4.36.1.4		InetZoneIndex	B	B
ipv6ScopeZoneIndexSiteLocal	1.3.6.1.2.1.4.36.1.5	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndex6	1.3.6.1.2.1.4.36.1.6	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndex7	1.3.6.1.2.1.4.36.1.7	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndexOrganizationLocal	1.3.6.1.2.1.4.36.1.8	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndex9	1.3.6.1.2.1.4.36.1.9	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndexA	1.3.6.1.2.1.4.36.1.10	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndexB	1.3.6.1.2.1.4.36.1.11	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndexC	1.3.6.1.2.1.4.36.1.12	RO	InetZoneIndex	B	B
ipv6ScopeZoneIndexD	1.3.6.1.2.1.4.36.1.13	RO	InetZoneIndex	B	B

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN performs a GetNext walk on NUT for IPv6 Scope Zone Index Table.

Judgment

Examine the return OID values for each B(mandatory) object identifier.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.7

v6SNMPv2CMIB2.8 Default Router Table

Purpose

Default Router Table lists the default routers known to this entity. This table is intended to be a simple list to display the information that end nodes may have been configured with or acquired through a simple system such as IPv6 router advertisements. Managers attempting to view more complicated routing information should examine the routing specific tables from other MIBs. This table is required for all entities.

This test shall check the existence of Default Router Table and the values of the OIDs in this table which include ipDefaultRouterIfIndex, ipDefaultRouterLifetime and ipDefaultRouterPreference. Table 11 lists the test criteria for this IP Default Router Table test. B stands for mandatory objects and they shall be used for judgment criteria.

Table 11 RFC 4293 IP MIB – IP Default Router Table Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ipDefaultRouterTable	1.3.6.1.2.1.4.37	NA	SEQUENCE OF IpDefaultRouterEntry	B	B
ipDefaultRouterEntry	1.3.6.1.2.1.4.37.1	NA	IpDefaultRouterEntry	B	B
ipDefaultRouterAddressType	1.3.6.1.2.1.4.37.1.1	NA	InetAddressType {ipv4(1), ipv6(2), ipv4z(3), ipv6z(4)}	B	B
ipDefaultRouterAddress	1.3.6.1.2.1.4.37.1.2	NA	InetAddress (Size(4 | 8 | 16 | 20))	B	B
ipDefaultRouterIfIndex	1.3.6.1.2.1.4.37.1.3	RO	InterfaceIndex	B	B
ipDefaultRouterLifetime	1.3.6.1.2.1.4.37.1.4	RO	Unsigned32(0..65535)	B	B
ipDefaultRouterPreference	1.3.6.1.2.1.4.37.1.5	RO	INTEGER {reserved (-2), low (-1), medium (0), high (1) }	B	B

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN performs a GetNext walk on NUT for IP Default Router Table.

Judgment

Examine the return OID values for each B(mandatory) object identifier for valid syntax type and value range.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.8

v6SNMPv2CMIB2.9 IPv6 Router Advertisement Table

Purpose

Router Advertisement Table contains the non-routing information that an IPv6 router would use in constructing a router advertisement message. It does not contain information about the prefixes or other routing specific information that the router might advertise. The router should acquire such information from either the routing tables or from some routing table specific MIB. This table is only required for IPv6 router entities.

This test shall verify the value of IPv6 Router Advertisement Table correctly. Table 12 is the test criteria for IPv6 Router Advertisement Table test. B stands for Basic(mandatory) objects and they shall be used for judgment criteria.

Table 12 RFC 4293 IP MIB – IPv6 Router Advertisement Table Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
ipv6RouterAdvertTable (Routers only)	1.3.6.1.2.1.4.39	NA	SEQUENCE OF ipv6RouterAdvertEntry	-	B
ipv6RouterAdvertEntry	1.3.6.1.2.1.4.39.1	NA	ipv6RouterAdvertEntry	-	B
ipv6RouterAdvertIfIndex	1.3.6.1.2.1.4.39.1.1	NA	InterfaceIndex	-	B
ipv6RouterAdvertSendAdverts	1.3.6.1.2.1.4.39.1.2	RC	Truthvalue	-	B
ipv6RouterAdvertMaxInterval	1.3.6.1.2.1.4.39.1.3	RC	Unsigned32 (4..1800)	-	B
ipv6RouterAdvertMinInterval	1.3.6.1.2.1.4.39.1.4	RC	Unsigned32 (3..1350)	-	B
ipv6RouterAdvertManagedFlag	1.3.6.1.2.1.4.39.1.5	RC	Truthvalue	-	B
ipv6RouterAdvertOtherConfigFlag	1.3.6.1.2.1.4.39.1.6	RC	Truthvalue	-	B
ipv6RouterAdvertLinkMTU	1.3.6.1.2.1.4.39.1.7	RC	Unsigned32	-	B
ipv6RouterAdvertReachableTime	1.3.6.1.2.1.4.39.1.8	RC	Unsigned32 (0..3600000)	-	B
ipv6RouterAdvertRetransmitTime	1.3.6.1.2.1.4.39.1.9	RC	Unsigned32	-	B
ipv6RouterAdvertCurHopLimit	1.3.6.1.2.1.4.39.1.10	RC	Unsigned32 (0..255)	-	B
ipv6RouterAdvertDefaultLifetime	1.3.6.1.2.1.4.39.1.11	RC	Unsigned32 (0|4..9000)	-	B
ipv6RouterAdvertRowStatus	1.3.6.1.2.1.4.39.1.12	RC	RowStatus	-	B

Resource Requirements

- Packet generator

- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN performs GetNext walk on NUT for IPv6 Router Advertisement Table.

Judgment

Examine the return OID values for each B(mandatory) object identifier for valid syntax type and value range.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.9

v6SNMPv2CMIB2.10 ICMP Statistics Table

Purpose

ICMP Statistics Tables include two sets of statistics for ICMP. The first contains a simple set of counters to track the number of ICMP messages and errors processed by this entity. The second supplies more detail about the ICMP messages processed by this entity. Both of these tables are required for all entities.

This test shall check the existence of ICMP Statistics Table and verify the values for the OIDs in this table. Table 13 is the test criteria for this ICMP Statistics Table test. B (mandatory) objects shall be used for judgment criteria.

Table 13 RFC 4293 IP MIB – ICMP Statistics Table Test Criteria

Name	OID	MAX-Access	Syntax	Host	Router
icmpStatsTable	1.3.6.1.2.1.5.29	NA	SEQUENCE OF IcmpStatsEntry	B	B
icmpStatsEntry	1.3.6.1.2.1.5.29.1	NA	IcmpStatsEntry	B	B
icmpStatsIPversion	1.3.6.1.2.1.5.29.1.1	NA	Inetversion {ipv4(1), ipv6(2)}	B	B
icmpStatsInMsgs	1.3.6.1.2.1.5.29.1.2	RO	Counter32	B	B
icmpStatsInErrors	1.3.6.1.2.1.5.29.1.3	RO	Counter32	B	B
icmpStatsOutMsgs	1.3.6.1.2.1.5.29.1.4	RO	Counter32	B	B
icmpStatsOutErrors	1.3.6.1.2.1.5.29.1.5	RO	Counter32	B	B
icmpMsgStatsTable	1.3.6.1.2.1.5.30	NA	SEQUENCE OF IcmpMsgStatsEntry	B	B
icmpMsgStatsEntry	1.3.6.1.2.1.5.30.1	NA	IcmpMsgStatsEntry	B	B
icmpMsgStatsIPversion	1.3.6.1.2.1.5.30.1.1	NA	InetVersion {ipv4(1), ipv6(2)}	B	B
icmpMsgStatsType	1.3.6.1.2.1.5.30.1.2	NA	Integer32(0..255)	B	B
icmpMsgStatsInPkts	1.3.6.1.2.1.5.30.1.3	RO	Counter32	B	B
icmpMsgStatsOutPkts	1.3.6.1.2.1.5.30.1.4	RO	Counter32	B	B

Resource Requirements

- Packet generator
- Monitor to capture packets

Initialization

Network Topology

Please refer Fig 5. Test Architecture.

Setup

Refer Common Test Setup

Procedure

TN performs GetNext walk on NUT for icmpStatsTable and icmpMsgStatsTable

Judgment

Returned OID values for each B(mandatory) object identifier are with valid syntax type and within defined value range.

References

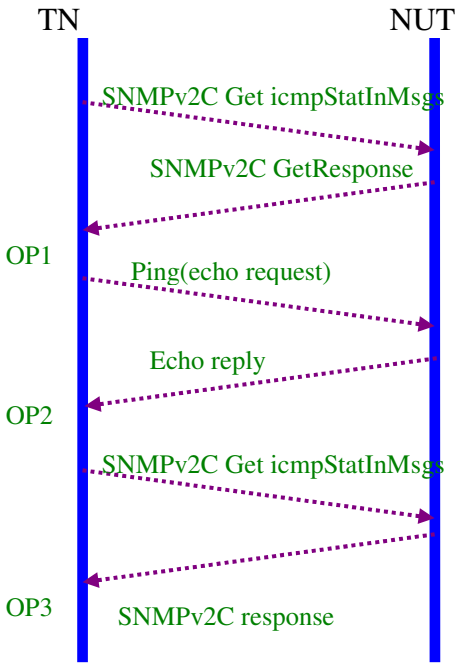
RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.10

v6SNMPv2CMIB2.10.1 icmpStatInMsgs counter check

Purpose

This test shall check and verify the values for the icmpStatInMsgs counter in ICMP Statistics Table using ping operation.

Procedure



1. TN sends SNMPv2C GetRequest with icmpStatInMsgs before the ping operation.
2. NUT replies SNMPv2C Response with the icmpStatInMsgs counter values.
3. TN ping(echo request) NUT
4. NUT replies(echo reply) to TN
5. TN sends SNMPv2C GetRequest with icmpStatInMsgs again after the ping operation.
6. NUT replies SNMPv2C Response with the icmpStatInMsgs counter values.

Judgment

OP1: TN receives icmpStatInMsgs counter value from NUT(icmpStatInMsgsCounter1).
OP2: TN receives echo reply from NUT.
OP3: TN receives icmpStatInMsgs counter value from NTU(icmpStatInMsgsCounter2) is correctly incremented by one, i.e. icmpStatInMsgsCounter2 = icmpStatInMsgsCounter1 + 1.

References

RFC 3416, Protocol Operations for version 2 of the Simple Network Management Protocol
RFC 4293, Management Information Base for the Internet Protocol (IP), Sec 3.2.10