MODIFICATION RECORD

Version 2.0.0 May.10, 2011
Major Version Up Items
- Change EN.I.1.2.4.1
- Change SGW.I.1.2.4.1
- Add [EN.R.P29.L2503.ADD]
- Add [SGW.R.P29.L2503.ADD]
- Add [EN.R.P86.L4030.ADD.1]
- Add [SGW.R.P86.L4030.ADD.1]
- Add [EN.R.P86.L4030.ADD.2]
- Add [SGW.R.P86.L4030.ADD.2]
- Add [EN.R.P86.L4034.ADD.1]
- Add [SGW.R.P86.L4034.ADD.1]
- Add [EN.R.P86.L4034.ADD.2]
- Add [SGW.R.P86.L4034.ADD.2]
- Add [EN.R.P86.L4034.ADD.3]
- Add [SGW.R.P86.L4034.ADD.3]
- Add [EN.R.P86.L4034.ADD.4]
- Add [SGW.R.P86.L4034.ADD.4]
- Add [EN.R.P57.L2663.ADD]
- Add [SGW.R.P57.L2663.ADD]
- Add [EN.R.P69.L3234.ADD]
- Add [SGW.R.P69.L3234.ADD]
- Add [EN.R.P69.L3252.ADD]
- Add [SGW.R.P69.L3252.ADD]
- Add [EN.R.P69.L3258.ADD]
- Add [SGW.R.P69.L3258.ADD]
- Add [EN.R.P69.L3260.ADD]
- Add [SGW.R.P69.L3260.ADD]
- Add [SGW.R.P116.L5437.ADD]

Version 1.1.0 Jun. 8, 2010
Major Revision Up Items
- IKEv2.{EN,SGW}.{I,R}.1.1.6.1 Part F - Supported PRF PRF_HMAC_SHA2_256
- IKEv2.{EN,SGW}.{I,R}.1.1.6.1 Part G - Supported Integrity Algorithm AUTH_HMAC_SHA2_256 for IKE SA
- IKEv2.{EN,SGW}.R.1.1.6.3 - Changed to choose Diffie-Hellman Group 14 or Diffie-Hellman Group 24
- IKEv2.{EN,SGW}.R.1.1.6.4 - Changed to choose Diffie-Hellman Group 14 or Diffie-Hellman Group 24
- IKEv2.{EN,SGW}.R.1.1.6.4 Part A-D - Allowed only Notify type of UNSUPPORTED_CRITICAL_PAYLOAD
- IKEv2.{EN,SGW}.R.1.1.6.8 - Changed to receive Notify type of NO_PROPOSAL_CHOSEN
- IKEv2.{EN,SGW}.R.1.1.6.9 - Added IKE_SA Rekeying Failure test cases

Minor Revision Up Items
- IKEv2.{EN,SGW}.R.1.1.6.9 - Added IKE_SA Rekeying Failure test cases
- IKEv2.{EN,SGW}.R.1.1.4.4 Part A-D - Allowed only Notify type of UNSUPPORTED_CRITICAL_PAYLOAD
IKEv2.{EN,SGW}.I.1.1.5.2 - Removed test cases for COOKIE generation because of untestable test case
IKEv2.{EN,SGW}.I.1.1.5.3 - Removed test cases for COOKIE generation because of untestable test case
IKEv2.{EN,SGW}.I.1.1.5.4 - Removed test cases for COOKIE generation because of untestable test case
IKEv2.{EN,SGW}.I.1.2.8.1 - Removed test cases for AUTHENTICATION_FAILED because of untestable test case
IKEv2.{EN,SGW}.I.1.1.6.1 Part B - Removed test cases using AES CTR for IKE_SA negotiation
IKEv2.{EN,SGW}.I.1.1.6.1 Part B - Removed test cases using AES CTR for IKE_SA negotiation
IKEv2.{EN,SGW}.I.1.2.4.7 - Fixed typo
IKEv2.EN.I.2.1.2.[2,3,4,5] - Added Possible Problems
IKEv2.{EN,SGW}.I.1.1.6.12 - Changed to be more realistic test sequence
IKEv2.{EN,SGW}.I.1.1.3.5 - Removed test cases for sending liveness check because of untestable
IKEv2.{EN,SGW}.I.1.3.1.1 - Removed test cases for sending liveness check because of untestable
IKEv2.{EN,SGW}.I.1.3.2.1 - Removed test cases for sending liveness check because of untestable
IKEv2.{EN,SGW}.I.1.3.2.2 - Removed test cases for sending liveness check because of untestable
IKEv2.{EN,SGW}.I.1.3.3.1 - Removed test cases for sending liveness check because of untestable
IKEv2.{EN,SGW}.I.1.1.3.8 - Removed test cases for sending liveness check because of untestable
IKEv2.{EN,SGW}.I.1.2.6.1 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.2 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.3 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.5.6 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.6.5 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.6.6 - Fixed typo
IKEv2.{EN,SGW}.I.1.3.4.2 - Fixed typo
IKEv2.{EN,SGW}.I.1.2.3.6 - Changed to use rekeying IKE_SA instead of rekeying CHILD_SA
IKEv2.{EN,SGW}.I.1.2.3.7 - Changed to use rekeying IKE_SA instead of rekeying CHILD_SA
IKEv2.{EN,SGW}.I.1.2.6.7 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.8 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.9 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.10 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.11 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.12 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.13 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.14 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.15 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.3.7 - Removed test cases for CHILD_SA deletion because of untestable
IKEv2.SGW.I.1.1.6.10 - Fixed typo
IKEv2.{EN,SGW}.I.1.2.3.6 - Changed to use rekeying IKE_SA instead of rekeying CHILD_SA
IKEv2.{EN,SGW}.R.1.2.6.8 - Changed to use rekeying IKE_SA instead of rekeying CHILD_SA
IKEv2.{EN,SGW}.I.1.2.3.8 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.5.6 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.6.5 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.6.7 - Removed test cases using AES_CTR for IKE_SA negotiation
IKEv2.{EN,SGW}.I.1.2.6.8 - Removed test cases using AES_CTR for IKE_SA negotiation
IKEv2.{EN,SGW}.I.1.2.6.9 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.10 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.11 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.12 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.13 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.14 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.2.6.15 - Removed test cases for exchange collision because of untestable
IKEv2.{EN,SGW}.I.1.3.7 - Removed test cases for CHILD_SA deletion because of untestable
IKEv2.SGW.I.1.1.6.10 - Fixed typo
IKEv2.{EN,SGW}.I.1.2.3.6 - Changed to use rekeying IKE_SA instead of rekeying CHILD_SA
IKEv2.{EN,SGW}.R.1.2.6.8 - Changed to use rekeying IKE_SA instead of rekeying CHILD_SA
IKEv2.{EN,SGW}.I.1.2.3.8 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.5.6 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.6.5 - Changed to allow both the new CHILD_SA and the old CHILD_SA
IKEv2.{EN,SGW}.I.1.2.6.7 - Changed requirements from BASIC to ADVANCED since these tests requires NUT to transmit multiple transforms and to support 2048 MODP Group
IKEv2.{EN,SGW}.I.1.1.6.11 - Changed requirements from BASIC to ADVANCED since these tests requires NUT to transmit multiple transforms, to support 2048 MODP Group and to support PFS
IKEv2.{EN,SGW}.I.1.2.3.7, IKEv2.{EN,SGW}.R.1.2.8.1 Part C - Changed requirements from BASIC to ADVANCED since these tests requires NUT to support PFS

Version 1.0.3
Sep. 14, 2009
IKEv2.{EN,SGW}.I.1.1.6.2 Part E - Permitted to omit transform when the integrity algorithm is NONE
IKEv2.{EN,SGW}.I.1.1.5.[2-3], IKEv2.{EN,SGW}.I.1.1.6.[7,11], IKEv2.{EN,SGW}.R.1.1.5.[3-4], IKEv2.{EN,SGW}.I.1.1.6.[7-8] - Updated INVALID_KE_PAYLOAD test procedure to be realistic
IKEv2.{EN,SGW}.I.1.1.6.7 - Mandated to transmit INVALID_ID_KEY_PAYLOAD since it is required as MUST in RFC 4306
IKEv2.{EN,SGW}.I.1.1.6.7 - Changed requirements from BASIC to ADVANCED since these tests requires NUT to transmit multiple transforms and to support 2048 MODP Group
IKEv2.{EN,SGW}.I.1.1.6.11 - Changed requirements from BASIC to ADVANCED since these tests requires NUT to transmit multiple transforms, to support 2048 MODP Group and to support PFS
IKEv2.{EN,SGW}.I.1.2.3.7, IKEv2.{EN,SGW}.R.1.2.8.1 Part C - Changed requirements from BASIC to ADVANCED since these tests requires NUT to support PFS

Version 1.0.2
Jun. 02, 2009
Requirements - Unsupport send / receive ID_IPV4_ADDR / ID_FQDN / ID_RFC822_ADDR function by mandating to support ID_IPV6_ADDR
{EN,SGW}.I.1.1.9.1, {EN,SGW}.I.1.1.9.2, {EN,SGW}.R.1.1.9.1, {EN,SGW}.R.1.1.9.2 - Remove send / receive ID_IPV4_ADDR / ID_FQDN / ID_RFC822_ADDR test cases by mandating to support ID_IPV6_ADDR
Function List, {EN,SGW}.I.1.2.5.2 - Clarify Additional CHILD_SA function is ADVANCED
EN.R.1.1.7.2 - Fix editorial typo
{EN,SGW}.R.1.3.1.1 - Correct test Purpose
{EN,SGW}.I.1.2.3.6 - Fix editorial typo
EN.I.2.1.1.1, EN.I.2.1.1.2, EN.R.2.1.1.1, EN.R.2.1.1.2 - Fix editorial typo

Version 1.0.1
Apr. 15, 2009
IKEv2.EN.I.1.1.5.2, IKEv2.SGW.I.1.1.5.2, IKEv2.EN.R.1.1.5.3, IKEv2.SGW.R.1.1.5.3, IKEv2.EN.R.1.1.5.4, IKEv2.SGW.R.1.1.5.4 - Update acceptable packets and check establishment of IKE_SA
IKEv2.EN.I.1.1.5.3, IKEv2.SGW.I.1.1.5.3 - Add new test cases for Intetaction of COOKIE and INVALID_ID_KEY_PAYLOAD with unoptimized Responder

Version 1.0.0
Dec. 11, 2008
Initial release

IPv6 FORUM TECHNICAL DOCUMENT IPv6 Ready Logo Program IKEv2
ACKNOWLEDGMENTS

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

Authors:

Yokogawa Electric Corporation
Nippon Telegraph and Telephone Corporation (NTT)

Commentators:

NTT Advanced Technology Corporation (NTT-AT)
University of New Hampshire - InterOperability Lab
IRISA-INRIA

Note:

Development of this document was supported in part by a grant from NICT.
INTRODUCTION

Overview

TAHI Project is the joint effort formed with the objective of developing and providing the verification technology for IPv6.

The growth process of IPv4 was the history of encountering various kinds of obstacles and conquering such obstacles. However, once the position as infrastructure was established, it is not allowed to repeat the same history. This is a reason why the verification technology is essential for IPv6 deployment.

We research and develop conformance tests and interoperability tests for IPv6.

We closely work with the KAME project and USAGI project. We help activities of these projects in the quality side by offering the verification technology we develop in TAHI project and improve the development efficiency.

We open the results and fruits of the project to the public for FREE. Any developer concerned with IPv6 can utilize the results and fruits of TAHI project freely. Free software plays an important role in progress of the Internet. We believe that providing the verification technology for FREE contributes to advances of IPv6. Besides the programs, the specifications and criteria of verification will be included in the Package.

Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TN</td>
<td>Testing Node</td>
</tr>
<tr>
<td>TH</td>
<td>Testing Host</td>
</tr>
<tr>
<td>TR</td>
<td>Testing Router</td>
</tr>
<tr>
<td>NUT</td>
<td>Node Under Test</td>
</tr>
<tr>
<td>HUT</td>
<td>Host Under Test</td>
</tr>
<tr>
<td>RUT</td>
<td>Router Under Test</td>
</tr>
<tr>
<td>IKE</td>
<td>Internet Key Exchange (IKEv2) Protocol</td>
</tr>
<tr>
<td>EN</td>
<td>End-Node</td>
</tr>
<tr>
<td>SGW</td>
<td>Security-Gateway</td>
</tr>
<tr>
<td>PSK</td>
<td>Pre-Shared Key</td>
</tr>
<tr>
<td>AUTH</td>
<td>Authentication Payload</td>
</tr>
<tr>
<td>CERT</td>
<td>Certificate Payload</td>
</tr>
<tr>
<td>CERTREQ</td>
<td>Certificate Request Payload</td>
</tr>
<tr>
<td>CP</td>
<td>Configuration Payload</td>
</tr>
<tr>
<td>D</td>
<td>Delete Payload</td>
</tr>
<tr>
<td>E</td>
<td>Encrypted Payload</td>
</tr>
<tr>
<td>EAP</td>
<td>Extensible Authentication Payload</td>
</tr>
<tr>
<td>HDR</td>
<td>IKE Header</td>
</tr>
<tr>
<td>IDi</td>
<td>Identification - Initiator Payload</td>
</tr>
<tr>
<td>IDR</td>
<td>Identification - Responder Payload</td>
</tr>
<tr>
<td>KE</td>
<td>Key Exchange Payload</td>
</tr>
<tr>
<td>Ni</td>
<td>Nonce - Initiator Payload</td>
</tr>
<tr>
<td>Nr</td>
<td>Nonce - Responder Payload</td>
</tr>
<tr>
<td>N</td>
<td>Notify Payload</td>
</tr>
<tr>
<td>SA</td>
<td>Security Association Payload</td>
</tr>
<tr>
<td>TSi</td>
<td>Traffic Selector - Initiator Payload</td>
</tr>
<tr>
<td>TSr</td>
<td>Traffic Selector - Responder Payload</td>
</tr>
<tr>
<td>V</td>
<td>Vendor ID Payload</td>
</tr>
</tbody>
</table>
TEST ORGANIZATION

This document organizes tests by Section 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:

Test Label: The test label and title comprise the first line of the test block. The test label is composed by concatenating the short test suite name, the section number, the group number, and the test number within the group. These elements are separated by periods. The Test Number is the section, group and test number, also separated by periods.

Purpose: The Purpose is a short statement describing what the test attempts to achieve. It is usually phrased as a simple assertion of the feature or capability to be tested.

References: The References section lists cross-references to the specifications and documentation that might be helpful in understanding and evaluating the test and results.

Resource Requirements: The Resource Requirements section specifies the software, hardware, and test equipment that will be needed to perform the test.

Test Setup: The Test Setup section describes the configuration of all devices prior to the start of the test. Different parts of the procedure may involve configuration steps that deviate from what is given in the test setup. If a value is not provided for a protocol parameter, then the protocol’s default is used for that parameter.

Procedure: This section of the test description contains the step-by-step instructions for carrying out the test. These steps include such things as enabling interfaces, unplugging devices from the network, or sending packets from a test station. The test procedure also cues the tester to make observations, which are interpreted in accordance with the observable results given for that test part.

Observable Results: This section lists observable results that can be examined by the tester to verify that the NUT is operating properly. When multiple observable results are possible, this section provides a short discussion on how to interpret them. The determination of a pass or fail for each test is usually based on how the NUT’s behavior compares to the results described in this section.

Possible Problems: This section contains a description of known issues with the test procedure, which may affect test results in certain situations.
REFERENCES

The following documents are referenced in this text:

- RFC 4307 - Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2), December, 2005
- RFC 4718 - IKEv2 Clarifications and Implementation Guidelines, October, 2006
# TABLE OF CONTENTS

**MODIFICATION RECORD** ................................................................................................................................. 1  
**ACKNOWLEDGMENTS** ........................................................................................................................................ 3  
**INTRODUCTION** ................................................................................................................................................... 4  
**TEST ORGANIZATION** ........................................................................................................................................... 5  
**REFERENCES** ......................................................................................................................................................... 6  
**TABLE OF CONTENTS** ............................................................................................................................................. 7  

**Requirements** ....................................................................................................................................................... 18  
  - **EQUIPMENT TYPE** ........................................................................................................................................... 18  
  - **FUNCTION LIST** .............................................................................................................................................. 18  

**Common Topology** ............................................................................................................................................... 20  
  - **COMMON TOPOLOGY FOR END-NODE: END-NODE TO END-NODE** .............................................................. 20  
  - **COMMON TOPOLOGY FOR END-NODE: END-NODE TO SGW** ............................................................................. 21  
  - **COMMON TOPOLOGY FOR SGW: SGW TO SGW** .............................................................................................. 22  
  - **COMMON TOPOLOGY FOR SGW: SGW TO END-NODE** ..................................................................................... 23  

**Common Configuration for NUT** .......................................................................................................................... 24  
  - **COMMON CONFIGURATION FOR END-NODE: END-NODE TO END-NODE** ...................................................... 24  
  - **COMMON CONFIGURATION FOR END-NODE: END-NODE TO SGW** ............................................................... 25  
  - **COMMON CONFIGURATION FOR SGW: SGW TO SGW** .................................................................................... 26  
  - **COMMON CONFIGURATION FOR SGW: SGW TO END-NODE** ........................................................................ 27  

**Common Packets** ............................................................................................................................................... 28  
  - **IKE_SA_INIT MESSAGES** ................................................................................................................................ 28  
    - **Common Packet #1: IKE_SA_INIT request** ........................................................................................................... 28  
    - **Common Packet #2: IKE_SA_INIT response** ........................................................................................................ 30  
  - **IKE_AUTH MESSAGES** ......................................................................................................................................... 32  
    - **Common Packet #3: IKE_AUTH request for Transport Mode** ............................................................................... 32  
    - **Common Packet #4: IKE_AUTH response for Transport Mode** .......................................................................... 34  
    - **Common Packet #5: IKE_AUTH request for Tunnel Mode** .................................................................................. 36  
    - **Common Packet #6: IKE_AUTH response for Tunnel Mode** ............................................................................ 39  
  - **CREATE_CHILD_SA MESSAGES FOR GENERATING CHILD_SA** ................................................................. 42  
    - **Common Packet #7: CREATE_CHILD_SA request for Generating CHILD_SA for Transport Mode** ............ 42  
    - **Common Packet #8: CREATE_CHILD_SA request for Generating CHILD_SA for Tunnel Mode** ............ 44  
    - **Common Packet #9: CREATE_CHILD_SA response for Generating CHILD_SA for Transport Mode** ........... 46  
    - **Common Packet #10: CREATE_CHILD_SA response for Generating CHILD_SA for Tunnel Mode** ....... 48  
  - **CREATE_CHILD_SA MESSAGES FOR REKEYING IKE_SA** .............................................................. 50  
    - **Common Packet #11: CREATE_CHILD_SA request for Rekeying IKE_SA** ..................................................... 50  
    - **Common Packet #12: CREATE_CHILD_SA response for Rekeying IKE_SA** ........................................... 52  
  - **CREATE_CHILD_SA MESSAGES FOR REKEYING CHILD_SA** ............................................................ 54  
    - **Common Packet #13: CREATE_CHILD_SA request for Rekeying CHILD_SA for Transport Mode** ......... 54  
    - **Common Packet #14: CREATE_CHILD_SA response for Rekeying CHILD_SA for Transport Mode** ...... 56  
    - **Common Packet #15: CREATE_CHILD_SA request for Rekeying CHILD_SA for Tunnel Mode** ............ 58  
    - **Common Packet #16: CREATE_CHILD_SA response for Rekeying CHILD_SA for Tunnel Mode** ....... 60  

**INFORMATIONAL MESSAGES** .......................................................................................................................... 62
IPv6 FORUM TECHNICAL DOCUMENT

Group 1. The Initial Exchanges ................................................................. 67

Section 1. End Node .................................................................................. 67

Section 1.1. Initiator .................................................................................. 67

Section 1.1.1. Endpoint-to-Endpoint Transport ............................................ 67

Group 1.1. Header and Payload Formats ...................................................... 68

Test IKEv2.EN.I.1.1.1.1: Sending IKE_SA_INIT request .......................... 68
Test IKEv2.EN.I.1.1.1.2: Sending IKE_AUTH request ............................... 74
Test IKEv2.EN.I.1.1.1.3: Use of CHILD_SA .............................................. 85

Group 1.2. Use of Retransmission Timers ................................................... 87

Test IKEv2.EN.I.1.1.2.1: Retransmissions of IKE_SA_INIT requests .......... 87
Test IKEv2.EN.I.1.1.2.2: Stop of retransmission of IKE_SA_INIT requests ................................. 89
Test IKEv2.EN.I.1.1.2.3: Retransmissions of IKE_AUTH requests ............. 91
Test IKEv2.EN.I.1.1.2.4: Stop of retransmission of IKE_AUTH requests .... 93

Group 1.3. State Synchronization and Connection Timeouts ....................... 95

Test IKEv2.EN.I.1.1.3.1: State Synchronization with ICMP messages ........ 95
Test IKEv2.EN.I.1.1.3.2: State Synchronization with IKE messages ....... 97
Test IKEv2.EN.I.1.1.3.3: Close connections when repeated attempts fail .... 100
Test IKEv2.EN.I.1.1.3.4: Close connections when receiving INITIAL_CONTACT ..... 102
Test IKEv2.EN.I.1.1.3.5: Sending Liveness check ....................................... 103
Test IKEv2.EN.I.1.1.3.6: Sending Delete Payload for IKE_SA .................. 104
Test IKEv2.EN.I.1.1.3.7: Sending Delete Payload for CHILD_SA ........... 106
Test IKEv2.EN.I.1.1.3.8: Sending Liveness check with unprotected messages 107

Group 1.4. Version Numbers and Forward Compatibility .......................... 108

Test IKEv2.EN.I.1.1.4.1: Unrecognized payload types and Critical bit is not set 108
Test IKEv2.EN.I.1.1.4.2: Unrecognized payload types and Critical bit is set ...... 114

Group 1.5. Cookies .................................................................................... 120

Test IKEv2.EN.I.1.1.5.1: Retrying IKE_SA_INIT request with a Notify payload of type COOKIE 120
Test IKEv2.EN.I.1.1.5.2: Interaction of COOKIE and INVALID_KE_PAYLOAD .................................................. 123
Test IKEv2.EN.I.1.1.5.3: Interaction of COOKIE and INVALID_KE_PAYLOAD with unoptimized Responder 127

Group 1.6. Cryptographic Algorithm Negotiation ...................................... 131

Test IKEv2.EN.I.1.1.6.1: Cryptographic Algorithm Negotiation for IKE_SA ................................................... 131
Test IKEv2.EN.I.1.1.6.2: Cryptographic Algorithm Negotiation for CHILD_SA .............................................. 135
Test IKEv2.EN.I.1.1.6.3: Sending Multiple Transforms for IKE_SA ........... 140
Test IKEv2.EN.I.1.1.6.4: Sending Multiple Proposals for IKE_SA ................ 142
Test IKEv2.EN.I.1.1.6.5: Sending Multiple Transforms for CHILD_SA ....... 144
Test IKEv2.EN.I.1.1.6.6: Sending Multiple Proposals for CHILD_SA ........ 147
Test IKEv2.EN.I.1.1.6.7: Receipt of INVALID_KE_PAYLOAD ............................. 149
Test IKEv2.EN.I.1.1.6.8: Receipt of NO_PROPOSAL_CHOSEN ................................................................. 152

IPv6 FORUM TECHNICAL DOCUMENT

8

IPv6 Ready Logo Program IKEv2
Test IKEv2.EN.I.1.1.16.9: Response with inconsistent SA proposal for IKE_SA ................................................................. 153
Test IKEv2.EN.I.1.1.16.10: Response with inconsistent proposal for CHILD_SA ................................................................. 155
Test IKEv2.EN.I.1.1.16.11: Receipt of INVALID_KE_PAYLOAD in Initial Exchange ................................................................. 158
Test IKEv2.EN.I.1.1.16.12: Creating an IKE_SA without a CHILD_SA ........................................................................... 160

GROUP 1.7. TRAFFIC SELECTOR NEGOTIATION .............................................................................................................. 162
Test IKEv2.EN.I.1.1.7.1: Narrowing the range of members of the set of traffic selectors ................................................................. 162

GROUP 1.8. ERROR HANDLING ............................................................................................................................................. 165
Test IKEv2.EN.I.1.1.8.1: INVALID_IKE_SPI .......................................................................................................................... 165
Test IKEv2.EN.I.1.1.8.2: INVALID_SELECTORS ................................................................................................................... 166

GROUP 1.10. AUTHENTICATION OF THE IKE_SA .................................................................................................................. 167
Test IKEv2.EN.I.1.1.10.1: Sending CERT Payload ................................................................................................................ 167
Test IKEv2.EN.I.1.1.10.2: Sending CERTREQ Payload ......................................................................................................... 170
Test IKEv2.EN.I.1.1.10.3: RSA Digital Signature .................................................................................................................. 172
Test IKEv2.EN.I.1.1.10.4: HEX string PSK ....................................................................................................................... 175

GROUP 1.11. INVALID VALUES .............................................................................................................................................. 177
Test IKEv2.EN.I.1.1.11.1: Non zero RESERVED fields in IKE_SA_INIT response .............................................................. 177
Test IKEv2.EN.I.1.1.11.2: Non zero RESERVED fields in IKE_AUTH response ................................................................. 179
Test IKEv2.EN.I.1.1.11.3: Version bit is set ........................................................................................................................ 181
Test IKEv2.EN.I.1.1.11.4: Unrecognized Notify Message Type of Error ............................................................................. 183
Test IKEv2.EN.I.1.1.11.5: Unrecognized Notify Message Type of Status ............................................................................ 185

Group 2. The CREATE_CHILD_SA Exchange ..................................................................................................................... 187

GROUP 2.1. HEADER AND PAYLOAD FORMATS ................................................................................................................... 187
Test IKEv2.EN.I.2.1.1.1: Sending CREATE_CHILD_SA request ........................................................................................... 187

GROUP 2.2. USE OF RETRANSMISSION TIMERS ..................................................................................................................... 200
Test IKEv2.EN.I.2.2.1.1: Retransmissions of CREATE_CHILD_SA requests ........................................................................ 200
Test IKEv2.EN.I.2.2.2.2: Stop of retransmission of CREATE_CHILD_SA requests ................................................................. 203

GROUP 2.3. REKEYING CHILD_SAS USING A CREATE_CHILD_SA EXCHANGE .............................................................................. 206
Test IKEv2.EN.I.2.3.1.1: Close the replaced CHILD_SA ........................................................................................................ 206
Test IKEv2.EN.I.2.3.2.2: Use of the new CHILD_SA ............................................................................................................... 209
Test IKEv2.EN.I.2.3.3.3: Lifetime of CHILD_SA expires ........................................................................................................ 213
Test IKEv2.EN.I.2.3.4.4: Sending Multiple Transform ........................................................................................................... 215
Test IKEv2.EN.I.2.3.5.5: Sending Multiple Proposal ............................................................................................................. 219
Test IKEv2.EN.I.2.3.6.6: Rekeying Failure ............................................................................................................................ 221
Test IKEv2.EN.I.2.3.7.7: Perfect Forward Secrecy ................................................................................................................ 223
Test IKEv2.EN.I.2.3.8.8: Use of the old CHILD_SA ................................................................................................................ 227

GROUP 2.4. REKEYING IKE_SAS USING A CREATE_CHILD_SA EXCHANGE ............................................................................... 230
Test IKEv2.EN.I.2.4.1.1: Close the replaced IKE_SA ............................................................................................................... 230
Test IKEv2.EN.I.2.4.2.2: Use of the new IKE_SA .................................................................................................................... 233
Test IKEv2.EN.I.2.4.3.3: Lifetime of IKE_SA expires ................................................................................................................ 236
Test IKEv2.EN.I.2.4.4.4: Sending Multiple Transform ........................................................................................................... 238
Test IKEv2.EN.I.2.4.5.5: Sending Multiple Proposal ............................................................................................................. 243
Test IKEv2.EN.I.2.4.6.6: Use of the old IKE_SA .................................................................................................................... 246
Test IKEv2.EN.I.2.4.7.7: Changing PRFs when rekeying the IKE_SA ..................................................................................... 249

GROUP 2.5. CREATING NEW CHILD_SAS WITH THE CREATE_CHILD_SA EXCHANGES ................................................................. 252
Test IKEv2.EN.I.2.5.1.1: Create new CHILD_SA by sending CREATE_CHILD_SA request ................................................................. 252
Test IKEv2.EN.I.2.5.2.2: Receipt of cryptographically valid message on the new_SA .................................................................... 255

GROUP 2.6. EXCHANGE COLLISIONS .................................................................................................................................... 260
Test IKEv2.EN.I.2.6.1.1: Simultaneous CHILD_SA Close ....................................................................................................... 260
Test IKEv2.EN.I.2.6.2.2: Simultaneous IKE_SA Close ................................................................................................................ 261
Test IKEv2.EN.I.2.6.3.3: Simultaneous CHILD_SA Rekeying .................................................................................................. 262
Test IKEv2.EN.I.2.6.4.4: Simultaneous CHILD_SA Rekeying with retransmission ................................................................. 267
Test IKEv2.EN.I.2.6.5.5: Simultaneous IKE_SA Rekeying ........................................................................................................... 271
Test IKEv2.EN.I.2.6.6.6: Simultaneous IKE_SA Rekeying with retransmission ........................................................................ 275
Test IKEv2.EN.I.2.6.7.7: Rekeying a CHILD_SA while Closing a CHILD_SA ........................................................................... 279
Test IKEv2.EN.I.1.2.6.8: Closing a New CHILD_SA .......................................................... 280
Test IKEv2.EN.I.1.2.6.9: Rekeying a New CHILD_SA ..................................................... 281
Test IKEv2.EN.I.1.2.6.10: Rekeying an IKE_SA with half-open CHILD_SAs ............... 282
Test IKEv2.EN.I.1.2.6.11: Rekeying a CHILD_SA while rekeying an IKE_SA ............. 283
Test IKEv2.EN.I.1.2.6.12: Rekeying an IKE_SA with half-closed CHILD_SAs .............. 284
Test IKEv2.EN.I.1.2.6.13: Closing a CHILD_SA while rekeying an IKE_SA ............... 285
Test IKEv2.EN.I.1.2.6.14: Closing an IKE_SA while rekeying an IKE_SA ................ 286
Test IKEv2.EN.I.1.2.6.15: Rekeying an IKE_SA while Closing an IKE_SA ................. 287

Test IKEv2.EN.I.1.2.7.1: Non zero RESERVED fields in CREATE_CHILD_SA response ...... 288

Group 3. The INFORMATIONAL Exchange ....................................................................... 291

Test IKEv2.EN.I.1.3.1.1: Sending INFORMATIONAL Exchange ....................................... 291
Test IKEv2.EN.I.1.3.2.1: Retransmission of INFORMATIONAL request ....................... 292
Test IKEv2.EN.I.1.3.2.2: Stop of retransmission of INFORMATIONAL request ............... 293
Test IKEv2.EN.I.1.3.3.1: Non zero RESERVED fields in INFORMATIONAL response ....... 294
Test IKEv2.EN.I.1.3.4.1: INVALID_SPI .............................................................................. 296

Section 1.1.2. Endpoint to Security Gateway Tunnel ....................................................... 297

Group 1. The Initial Exchanges .......................................................................................... 297

Test IKEv2.EN.I.1.1.1.1: Sending IKE_AUTH request ..................................................... 297
Test IKEv2.EN.I.1.1.1.2: Use of CHILD_SA ................................................................. 308
Test IKEv2.EN.I.1.1.2.1: Sending CFG_REQUEST ......................................................... 310
Test IKEv2.EN.I.1.1.2.2: Receipt of CFG_REPLY .......................................................... 313
Test IKEv2.EN.I.1.1.2.3: Non zero RESERVED fields in Configuration Payload .......... 316
Test IKEv2.EN.I.1.1.2.4: Receipt of IKE_AUTH response without CFG_REPLY ............ 319
Test IKEv2.EN.I.1.1.2.5: Receipt of unrecognized Configuration Attributes ............... 322

Section 1.2. Responder ..................................................................................................... 325

Section 1.2.1. Endpoint-to-Endpoint Transport ............................................................... 325

Group 1. The Initial Exchanges .......................................................................................... 325

Test IKEv2.EN.R.1.1.1.1: Sending IKE_SA_INIT response ........................................... 326
Test IKEv2.EN.R.1.1.1.2: Sending IKE_AUTH response ................................................. 332
Test IKEv2.EN.R.1.1.1.3: Use of CHILD_SA ................................................................. 343
Test IKEv2.EN.R.1.1.2.1: Receipt of retransmitted IKE_SA_INIT request .................... 345
Test IKEv2.EN.R.1.1.2.2: Receipt of retransmitted IKE_AUTH request ......................... 347
Test IKEv2.EN.R.1.1.3.1: State Synchronization with ICMP messages ...................... 349
Test IKEv2.EN.R.1.1.3.2: State Synchronization with IKE messages ......................... 351
Test IKEv2.EN.R.1.1.3.3: Close connections when receiving INITIAL_CONTACT .......... 354
Test IKEv2.EN.R.1.1.3.4: Receiving Liveness check ....................................................... 355
Test IKEv2.EN.R.1.1.3.5: Receiving Delete Payload for IKE_SA .................................. 357
Test IKEv2.EN.R.1.1.3.6: Receiving Delete Payload for CHILD_SA ............................. 360
Test IKEv2.EN.R.1.1.4.1: Receipt of a higher minor version number ............................ 363
Test IKEv2.EN.R.1.1.4.2: Receipt of a higher major version number ........................................ 365
Test IKEv2.EN.R.1.1.4.3: Unrecognized payload types and critical bit is not set ......................... 367
Test IKEv2.EN.R.1.1.4.4: Unrecognized payload types and critical bit is set ............................ 371
Test IKEv2.EN.R.1.1.4.5: Invalid Order Payloads ................................................................. 375

Group 1.5. Cookies ............................................................................................................. 376
Test IKEv2.EN.R.1.1.5.1: Cookies ....................................................................................... 376
Test IKEv2.EN.R.1.1.5.2: Invalid Cookies ........................................................................... 377
Test IKEv2.EN.R.1.1.5.3: Interaction of COOKIE and INVALID_KE_PAYLOAD .................. 378
Test IKEv2.EN.R.1.1.5.4: Interaction of COOKIE and INVALID_KE_PAYLOAD with unoptimized
Initiator .......................................................................................................................... 379

Group 1.6. Cryptographic Algorithm Negotiation ............................................................. 380
Test IKEv2.EN.R.1.1.6.1: Cryptographic Algorithm Negotiation for IKE_SA ..................... 380
Test IKEv2.EN.R.1.1.6.2: Cryptographic Algorithm Negotiation for CHILD_SA ................. 385
Test IKEv2.EN.R.1.1.6.3: Receiving Multiple Transforms for IKE_SA ................................. 391
Test IKEv2.EN.R.1.1.6.4: Receiving Multiple Proposals for IKE_SA ................................. 394
Test IKEv2.EN.R.1.1.6.5: Receiving Multiple Transforms for CHILD_SA ......................... 398
Test IKEv2.EN.R.1.1.6.6: Receiving Multiple Proposals for CHILD_SA ......................... 401
Test IKEv2.EN.R.1.1.6.7: Sending INVALID_KE_PAYLOAD ............................................. 405
Test IKEv2.EN.R.1.1.6.8: Sending INVALID_KE_PAYLOAD in Initial Exchange .................. 409
Test IKEv2.EN.R.1.1.6.9: Creating an IKE_SA without a CHILD_SA .................................. 412

Group 1.7. Traffic Selector Negotiation ............................................................................ 414
Test IKEv2.EN.R.1.1.7.1: Narrowing Traffic Selectors ....................................................... 414
Test IKEv2.EN.R.1.1.7.2: TS_UNACCEPTABLE ................................................................. 417
Test IKEv2.EN.R.1.1.7.3: Narrowing Traffic Selectors from multiple Traffic Selector ............ 420

Group 1.8. Error Handling ............................................................................................... 424
Test IKEv2.EN.R.1.1.8.1: INVALID_KE_SPI ...................................................................... 424
Test IKEv2.EN.R.1.1.8.2: INVALID_SYNTAX ..................................................................... 425
Test IKEv2.EN.R.1.1.8.3: INVALID_SELECTORS .......................................................... 426

Group 1.10. Authentication of the IKE_SA ....................................................................... 427
Test IKEv2.EN.R.1.1.10.1: Sending Certificate Payload .................................................... 427
Test IKEv2.EN.R.1.1.10.2: Sending Certificate Request Payload ................................. 430
Test IKEv2.EN.R.1.1.10.3: RSA Digital Signature .............................................................. 432
Test IKEv2.EN.R.1.1.10.4: HEX string PSK ...................................................................... 435

Group 1.11 Invalid Values ................................................................................................. 437
Test IKEv2.EN.R.1.1.11.1: Non zero RESERVED fields in IKE_SA_INIT request ............... 437
Test IKEv2.EN.R.1.1.11.2: Non zero RESERVED fields in IKE_AUTH request .................... 439
Test IKEv2.EN.R.1.1.11.3: Version bit is set ...................................................................... 441
Test IKEv2.EN.R.1.1.11.4: Response bit is set ................................................................... 442
Test IKEv2.EN.R.1.1.11.5: Unrecognized Notify Message Type ........................................ 443

Group 2. The CREATE_CHILD_SA Exchange ................................................................. 446

Group 2.1. Header and Payload Formats ......................................................................... 446
Test IKEv2.EN.R.1.2.1.1: Receipt of CREATE_CHILD_SA request .................................. 446

Group 2.2. Use of Retransmission Timers ....................................................................... 457
Test IKEv2.EN.R.1.2.2.1: Receipt of retransmitted CREATE_CHILD_SA request ............. 457

Group 2.3. State Synchronization and Connection Timeouts ............................................ 459
Test IKEv2.EN.R.1.2.3.1: Receiving Delete Payload for Multiple CHILD_SA ................ 459

Group 2.4. Cryptographic Algorithm Negotiation ............................................................ 463
Test IKEv2.EN.R.1.2.4.1: Sending NO_PROPOSAL_CHOSEN ........................................ 463

Group 2.5. Rekeying CHILD_SA Using a CREATE_CHILD_SA Exchange ....................... 466
Test IKEv2.EN.R.1.2.5.1: Close the replaced CHILD_SA ............................................... 466
Test IKEv2.EN.R.1.2.5.2: Use of the new CHILD_SA ..................................................... 469
Test IKEv2.EN.R.1.2.5.3: Receiving Multiple Transform .................................................. 472
Test IKEv2.EN.R.1.2.5.4: Receiving Multiple Proposal ...................................................... 476
Test IKEv2.EN.R.1.2.5.5: Perfect Forward Secrecy ................................................................. 480
Test IKEv2.EN.R.1.2.5.6: Use of the旧 CHILD_SA ................................................................. 484

**GROUP 2.6. REKEYING IKE_SAS USING A CREATE_CHILD_SA EXCHANGE** .................. 486
Test IKEv2.EN.R.1.2.6.1: Sending CREATE_CHILD_SA response ...................................... 486
Test IKEv2.EN.R.1.2.6.2: Receipt of cryptographically valid message on the old SA ........ 488
Test IKEv2.EN.R.1.2.6.3: Receipt of cryptographically valid message on the new SA ...... 490
Test IKEv2.EN.R.1.2.6.4: Close the replaced IKE_SA .......................................................... 492
Test IKEv2.EN.R.1.2.6.5: Receiving Multiple Transform ..................................................... 495
Test IKEv2.EN.R.1.2.6.6: Receiving Multiple Proposal ......................................................... 499
Test IKEv2.EN.R.1.2.6.7: Changing PRFs when rekeying the IKE_SA ............................... 504
Test IKEv2.EN.R.1.2.6.8: D-H transform NONE when rekeying the IKE_SA .................. 507
Test IKEv2.EN.R.1.2.6.9: Rekeying Failure ............................................................................ 508

**GROUP 2.7. CREATING NEW CHILD_SAS USING A CREATE_CHILD_SA EXCHANGE** .... 511
Test IKEv2.EN.R.1.2.7.1: Receipt of cryptographically valid message on the new SA ....... 511

**GROUP 2.8. ERROR HANDLING** ....................................................................................... 516
Test IKEv2.EN.R.1.2.8.1: AUTHENTICATION_FAILED ....................................................... 516

**GROUP 2.9. NON ZERO RESERVED FIELDS** .................................................................. 517
Test IKEv2.EN.R.1.2.9.1: Non zero RESERVED fields in CREATE_CHILD_SA request ....... 517

**GROUP 3. THE INFORMATIONAL EXCHANGE** ................................................................. 519
**GROUP 3.1. HEADER AND PAYLOAD FORMATS** .............................................................. 519
Test IKEv2.EN.R.1.3.1.1: Sending INFORMATIONAL response ........................................ 519
**GROUP 3.2. USE OF RETRANSMISSION TIMERS** ............................................................... 523
Test IKEv2.EN.R.1.3.2.1: Receipt of retransmitted INFORMATIONAL request .................. 523
**GROUP 3.3. NON ZERO RESERVED FIELDS** .................................................................. 526
Test IKEv2.EN.R.1.3.3.1: Non RESERVED fields in INFORMATIONAL request................ 526

**Section 1.2.2. Endpoint to Security Gateway Tunnel** ....................................................... 550

**Section 1. Initial Exchanges** ............................................................................................ 550
**GROUP 1.1. HEADER AND PAYLOAD FORMATS** ............................................................. 551
Test IKEv2.EN.R.1.1.1.1: Sending IKE_AUTH response ..................................................... 551
Test IKEv2.EN.R.1.1.1.2: Use of CHILD_SA ........................................................................ 560

**Section 2. Security Gateway** ......................................................................................... 562

**Section 2.1. Initiator** ....................................................................................................... 562

**Section 2.1.1. Security Gateway to Security Gateway Tunnel** ......................................... 562

**Group 1. The Initial Exchanges** ..................................................................................... 562
**GROUP 1.1. HEADER AND PAYLOAD FORMATS** ............................................................. 563
Test IKEv2.SGW.I.1.1.1.1: Sending IKE_SA_INIT request .................................................. 563
Test IKEv2.SGW.I.1.1.1.2: Sending IKE_AUTH request ....................................................... 569
Test IKEv2.SGW.I.1.1.1.3: Use of CHILD_SA ................................................................. 579

**GROUP 1.2. USE OF RETRANSMISSION TIMERS** ............................................................. 581
Test IKEv2.SGW.I.1.2.1.1: Retransmissions of IKE_SA_INIT requests ............................... 581
Test IKEv2.SGW.I.1.2.1.2: Stop of retransmission of IKE_SA_INIT requests .................. 583
Test IKEv2.SGW.I.1.2.1.3: Retransmissions of IKE_AUTH requests .................................. 585
Test IKEv2.SGW.I.1.2.1.4: Stop of retransmission of IKE_AUTH requests ....................... 587

**GROUP 1.3. STATE SYNCHRONIZATION AND CONNECTION TIMEOUTS** ....................... 589
Test IKEv2.SGW.I.1.3.1: State Synchronization with ICMP messages ............................. 589
Test IKEv2.SGW.I.1.3.2: State Synchronization with IKE messages ............................... 592
Test IKEv2.SGW.I.1.3.3: Close connections when repeated attempts fail ....................... 595
Test IKEv2.SGW.I.1.3.4: Close connections when receiving INITIAL_CONTACT ............. 597
Test IKEv2.SGW.I.1.3.5: Sending Liveness check ............................................................... 598
Test IKEv2.SGW.I.1.1.3.6: Sending Delete Payload for IKE_SA ................................................................. 599
Test IKEv2.SGW.I.1.1.3.7: Sending Delete Payload for CHILD_SA ......................................................... 601
Test IKEv2.SGW.I.1.1.3.8: Sending Liveness check with unprotected messages ........................................... 602

GROUP 1.4. VERSION NUMBERS AND FORWARD COMPATIBILITY .............................................................. 603
Test IKEv2.SGW.I.1.1.4.1: Unrecognized payload types and critical bit is not set ........................................ 603
Test IKEv2.SGW.I.1.1.4.2: Unrecognized payload types and critical bit is set ............................................. 610

GROUP 1.5. COOKIES ................................................................................................................................. 617
Test IKEv2.SGW.I.1.1.5.1: Retrying IKE_SA_INIT request with a Notify payload of type COOKIE ............... 617
Test IKEv2.SGW.I.1.1.5.2: Interaction of COOKIE and INVALID_KE_PAYLOAD ........................................ 620
Test IKEv2.SGW.I.1.1.5.3: Interaction of COOKIE and INVALID_KE_PAYLOAD with unoptimized Responder ........................................................................................................... 624

GROUP 1.6. CRYPTOGRAPHIC ALGORITHM NEGOTIATION ............................................................................. 628
Test IKEv2.SGW.I.1.1.6.1: Cryptographic Algorithm Negotiation for IKE_SA ........................................... 628
Test IKEv2.SGW.I.1.1.6.2: Cryptographic Algorithm Negotiation for CHILD_SA .................................... 632
Test IKEv2.SGW.I.1.1.6.3: Sending Multiple Transforms for IKE_SA ......................................................... 637
Test IKEv2.SGW.I.1.1.6.4: Sending Multiple Proposals for IKE_SA ............................................................. 639
Test IKEv2.SGW.I.1.1.6.5: Sending Multiple Transforms for CHILD_SA .................................................. 641
Test IKEv2.SGW.I.1.1.6.6: Sending Multiple Proposals for CHILD_SA ..................................................... 644
Test IKEv2.SGW.I.1.1.6.7: Receipt of INVALID_KE_PAYLOAD ................................................................. 646
Test IKEv2.SGW.I.1.1.6.8: Receipt of NO_PROPOSAL_CHOSEN ................................................................. 649
Test IKEv2.SGW.I.1.1.6.9: Response with inconsistent SA proposal for IKE_SA ........................................ 650
Test IKEv2.SGW.I.1.1.6.10: Response with inconsistent proposal for CHILD_SA ..................................... 652
Test IKEv2.SGW.I.1.1.6.11: Receipt of INVALID_KE_PAYLOAD in Initial Exchange ................................ 655
Test IKEv2.SGW.I.1.1.6.12: Creating an IKE_SA without a CHILD_SA ..................................................... 657

GROUP 1.7. TRAFFIC SELECTOR NEGOTIATION .............................................................................................. 659
Test IKEv2.SGW.I.1.1.7.1: Narrowing the range of members of the set of traffic selectors ............................. 659

GROUP 1.8. ERROR HANDLING .................................................................................................................. 662
Test IKEv2.SGW.I.1.1.8.1: INVALID_IKE_SPI ............................................................................................... 662
Test IKEv2.SGW.I.1.1.8.2: INVALID_SELECTORS ..................................................................................... 663

GROUP 1.10 AUTHENTICATION OF THE IKE_SA .......................................................................................... 664
Test IKEv2.SGW.I.1.1.10.1: Sending CERT Payload ................................................................................ 664
Test IKEv2.SGW.I.1.1.10.2: Sending CERTREQ Payload ........................................................................... 667
Test IKEv2.SGW.I.1.1.10.3: RSA Digital Signature .................................................................................... 669
Test IKEv2.SGW.I.1.1.10.4: HEX string PSK ............................................................................................ 673

GROUP 1.11 INVALID VALUES ......................................................................................................................... 675
Test IKEv2.SGW.I.1.1.11.1: Non zero RESERVED fields in IKE_SA_INIT response .................................. 675
Test IKEv2.SGW.I.1.1.11.2: Non zero RESERVED fields in IKE_AUTH response ...................................... 677
Test IKEv2.SGW.I.1.1.11.3: Version bit is set ............................................................................................... 679
Test IKEv2.SGW.I.1.1.11.4: Unrecognized Notify Message Type of Error .................................................. 681
Test IKEv2.SGW.I.1.1.11.5: Unrecognized Notify Message Type of Status ................................................ 683

Group 2. The CREATE_CHILD_SA Exchange ................................................................................................ 685

GROUP 2.1. HEADER AND PAYLOAD FORMATS ............................................................................................ 685
Test IKEv2.SGW.I.1.2.1.1: Sending CREATE_CHILD_SA request ............................................................. 685

GROUP 2.2. USE OF RETRANSMISSION TIMERS .............................................................................................. 699
Test IKEv2.SGW.I.1.2.2.1: Retransmissions of CREATE_CHILD_SA requests ............................................. 699
Test IKEv2.SGW.I.1.2.2.2: Stop of retransmission of CREATE_CHILD_SA requests ............................... 702

GROUP 2.3. REKEYING CHILD_SA USING A CREATE_CHILD_SA EXCHANGE .............................................. 705
Test IKEv2.SGW.I.1.2.3.1: Close the replaced CHILD_SA ........................................................................... 705
Test IKEv2.SGW.I.1.2.3.2: Use of the new CHILD_SA .................................................................................. 708
Test IKEv2.SGW.I.1.2.3.3: Lifetime of CHILD_SA expires ........................................................................ 712
Test IKEv2.SGW.I.1.2.3.4: Sending Multiple Transforms ........................................................................... 714
Test IKEv2.SGW.I.1.2.3.5: Sending Multiple Proposal ................................................................................ 718
Test IKEv2.SGW.I.1.2.3.6: Rekeying Failure ................................................................................................. 720
Test IKEv2.SGW.I.1.2.3.7: Perfect Forward Secrecy .......................................................... 723
Test IKEv2.SGW.I.1.2.3.8: Use of the old CHILD_SA ...................................................... 727

GROUP 2.4. REKEYING IKE_SAS USING A CREATE_CHILD_SA EXCHANGE .................. 730
Test IKEv2.SGW.I.1.2.4.1: Close the replaced IKE_SA .................................................... 730
Test IKEv2.SGW.I.1.2.4.2: Use of the new IKE_SA .......................................................... 733
Test IKEv2.SGW.I.1.2.4.3: Lifetime of IKE_SA expires .................................................... 736
Test IKEv2.SGW.I.1.2.4.4: Sending Multiple Transform .................................................... 738
Test IKEv2.SGW.I.1.2.4.5: Sending Multiple Proposal ...................................................... 743
Test IKEv2.SGW.I.1.2.4.6: Use of the old IKE_SA ............................................................ 746
Test IKEv2.SGW.I.1.2.4.7: Changing PRFs when rekeying the IKE_SA ......................... 749

GROUP 2.5. CREATING NEW CHILD_SAS WITH THE CREATE_CHILD_SA EXCHANGES ... 753
Test IKEv2.SGW.I.1.2.5.1: Create new CHILD_SA by sending CREATE_CHILD_SA request ... 753
Test IKEv2.SGW.I.1.2.5.2: Receipt of cryptographically valid message on the new SA ....... 756

GROUP 2.6. EXCHANGE COLLISIONS .................................................................................. 762
Test IKEv2.SGW.I.1.2.6.1: Simultaneous CHILD_SA Close ............................................ 762
Test IKEv2.SGW.I.1.2.6.2: Simultaneous IKE_SA Close .................................................. 763
Test IKEv2.SGW.I.1.2.6.3: Simultaneous CHILD_SA Rekeying ........................................ 764
Test IKEv2.SGW.I.1.2.6.4: Simultaneous CHILD_SA Rekeying with retransmission ........ 769
Test IKEv2.SGW.I.1.2.6.5: Simultaneous IKE_SA Rekeying ............................................. 774
Test IKEv2.SGW.I.1.2.6.6: Simultaneous IKE_SA Rekeying with retransmission ............ 778
Test IKEv2.SGW.I.1.2.6.7: Rekeying a CHILD_SA while Closing a CHILD_SA ............... 782
Test IKEv2.SGW.I.1.2.6.8: Closing a New CHILD_SA ..................................................... 783
Test IKEv2.SGW.I.1.2.6.9: Rekeying a New CHILD_SA .................................................... 784
Test IKEv2.SGW.I.1.2.6.10: Rekeying an IKE_SA with half-open CHILD_SAs .................. 785
Test IKEv2.SGW.I.1.2.6.11: Rekeying a CHILD_SA while rekeying an IKE_SA ............... 786
Test IKEv2.SGW.I.1.2.6.12: Rekeying an IKE_SA with half-closed CHILD_SAs ............... 787
Test IKEv2.SGW.I.1.2.6.13: Closing a CHILD_SA while rekeying an IKE_SA .................. 788
Test IKEv2.SGW.I.1.2.6.14: Closing an IKE_SA while rekeying an IKE_SA ..................... 789
Test IKEv2.SGW.I.1.2.6.15: Rekeying an IKE_SA while Closing an IKE_SA ..................... 790

GROUP 2.7. NON ZERO RESERVED FIELDS .................................................................... 791
Test IKEv2.SGW.I.1.2.7.1: Non zero RESERVED fields in CREATE_CHILD_SA response .... 791

GROUP 3. THE INFORMATIONAL EXCHANGE .................................................................. 794
Test IKEv2.SGW.I.1.3.1.1: Sending INFORMATIONAL Exchange ...................................... 794

GROUP 3.2. USE OF RETRANSMISSION TIMERS ............................................................... 795
Test IKEv2.SGW.I.1.3.2.1: Retransmission of INFORMATIONAL request ...................... 795
Test IKEv2.SGW.I.1.3.2.2: Stop of retransmission of INFORMATIONAL request .......... 796

GROUP 3.3. NON ZERO RESERVED FIELDS ..................................................................... 797
Test IKEv2.SGW.I.1.3.3.1: Non zero RESERVED fields in INFORMATIONAL response .... 797

GROUP 3.4. ERROR HANDLING ....................................................................................... 798
Test IKEv2.SGW.I.1.3.4.1: INVALID_SPI ......................................................................... 798

Section 2.1.2. ENDPOINT TO SECURITY GATEWAY TUNNEL ........................................ 799

GROUP 1. THE INITIAL EXCHANGES ............................................................................... 799
Test IKEv2.SGW.I.1.1.1.1: Sending IKE_AUTH request ..................................................... 799
Test IKEv2.SGW.I.1.1.1.2: Use of CHILD_SA ................................................................. 810

Section 2.2. RESPONDER ................................................................................................. 812

Section 2.2.1. SECURITY GATEWAY TO SECURITY GATEWAY TUNNEL ..................... 812

GROUP 1. THE INITIAL EXCHANGES ............................................................................... 812
Test IKEv2.SGW.I.1.1.2.3.7: Perfect Forward Secrecy .......................................................... 813
Test IKEv2.SGW.R.1.1.1.1: Sending IKE_SA_INIT response .......................................................... 813
Test IKEv2.SGW.R.1.1.1.2: Sending IKE_AUTH response ............................................................. 819
Test IKEv2.SGW.R.1.1.1.3: Use of CHILD_SA ............................................................................. 829

GROUP 1.2. USE OF RETRANSMISSION TIMERS .................................................................. 831
Test IKEv2.SGW.R.1.1.2.1: Receipt of retransmitted IKE_SA_INIT request ......................... 831
Test IKEv2.SGW.R.1.1.2.2: Receipt of retransmitted IKE_AUTH request ................................ 833

GROUP 1.3. STATE SYNCHRONIZATION AND CONNECTION TIMEOUTS ............................ 835
Test IKEv2.SGW.R.1.1.3.1: State Synchronization with ICMP messages .............................. 835
Test IKEv2.SGW.R.1.1.3.2: State Synchronization with IKE messages ...................................... 838
Test IKEv2.SGW.R.1.1.3.3: Close connections when receiving INITIAL_CONTACT .......... 841
Test IKEv2.SGW.R.1.1.3.4: Receiving Liveness check .............................................................. 842
Test IKEv2.SGW.R.1.1.3.5: Receiving Delete Payload for IKE_SA .......................................... 844
Test IKEv2.SGW.R.1.1.3.6: Receiving Delete Payload for CHILD_SA ..................................... 846

GROUP 1.4. VERSION NUMBERS AND FORWARD COMPATIBILITY ................................. 848
Test IKEv2.SGW.R.1.1.4.1: Receipt of a higher minor version number ..................................... 848
Test IKEv2.SGW.R.1.1.4.2: Receipt of a higher major version number ................................. 850
Test IKEv2.SGW.R.1.1.4.3: Unrecognized payload types and critical bit is not set ................ 852
Test IKEv2.SGW.R.1.1.4.4: Unrecognized payload types and critical bit is set ..................... 856
Test IKEv2.SGW.R.1.1.4.5: Invalid Order Payloads ................................................................. 860

GROUP 1.5. COOKIES ............................................................................................................ 861
Test IKEv2.SGW.R.1.1.5.1: Cookies .......................................................................................... 861
Test IKEv2.SGW.R.1.1.5.2: Invalid Cookies ............................................................................. 862
Test IKEv2.SGW.R.1.1.5.3: Interaction of COOKIE and INVALID_KE_PAYLOAD .............. 863
Test IKEv2.SGW.R.1.1.5.4: Interaction of COOKIE and INVALID_KE_PAYLOAD with unoptimized Initiator ................................................................. 864

GROUP 1.6. CRYPTOGRAPHIC ALGORITHM NEGOTIATION .............................................. 865
Test IKEv2.SGW.R.1.1.6.1: Cryptographic Algorithm Negotiation for IKE_SA .................. 865
Test IKEv2.SGW.R.1.1.6.2: Cryptographic Algorithm Negotiation for CHILD_SA ................ 870
Test IKEv2.SGW.R.1.1.6.3: Receiving Multiple Transforms for IKE_SA ............................... 877
Test IKEv2.SGW.R.1.1.6.4: Receiving Multiple Proposals for IKE_SA ................................. 880
Test IKEv2.SGW.R.1.1.6.5: Receiving Multiple Transforms for CHILD_SA ............................. 884
Test IKEv2.SGW.R.1.1.6.6: Receiving Multiple Proposals for CHILD_SA ............................... 887
Test IKEv2.SGW.R.1.1.6.7: Sending INVALID_KE_PAYLOAD ................................................ 891
Test IKEv2.SGW.R.1.1.6.8: Sending INVALID_KE_PAYLOAD in Initial Exchange ............... 895
Test IKEv2.SGW.R.1.1.6.9: Creating an IKE_SA without a CHILD_SA ................................. 898

GROUP 1.7. TRAFFIC SELECTION NEGOTIATION .......................................................... 900
Test IKEv2.SGW.R.1.1.7.1: Narrowing Traffic Selectors ......................................................... 900
Test IKEv2.SGW.R.1.1.7.2: TS_UNACCEPTABLE ...................................................................... 903
Test IKEv2.SGW.R.1.1.7.3: Narrowing Traffic Selectors ......................................................... 906

GROUP 1.8. ERROR HANDLING ......................................................................................... 910
Test IKEv2.SGW.R.1.1.8.1: INVALID_KE_SPI ........................................................................ 910
Test IKEv2.SGW.R.1.1.8.2: INVALID_SYNTAX ....................................................................... 911
Test IKEv2.SGW.R.1.1.8.3: INVALID_SELECTORS ................................................................. 912

GROUP 1.9 AUTHENTICATION OF THE IKE_SA .................................................................. 913
Test IKEv2.SGW.R.1.1.9.10.1: Sending Certificate Payload .................................................... 913
Test IKEv2.SGW.R.1.1.9.10.2: Sending Certificate Request Payload ....................................... 916
Test IKEv2.SGW.R.1.1.9.10.3: RSA Digital Signature .............................................................. 918
Test IKEv2.SGW.R.1.1.9.10.4: HEX string PSK ....................................................................... 922

GROUP 1.10 INVALID VALUES ............................................................................................. 924
Test IKEv2.SGW.R.1.1.11.1: Non zero RESERVED fields in IKE_SA_INIT request .............. 924
Test IKEv2.SGW.R.1.1.11.2: Non zero RESERVED fields in IKE_AUTH request ..................... 926
Test IKEv2.SGW.R.1.1.11.3: Version bit is set ......................................................................... 928
Test IKEv2.SGW.R.1.1.11.4: Response bit is set ..................................................................... 929
Test IKEv2.SGW.R.1.1.11.5: Unrecognized Notify Message Type ......................................... 930
Group 2. The CREATE_CHILD_SA Exchange

GROUP 2.1. HEADER AND PAYLOAD FORMATS................................................................. 933
Test IKEv2.SGW.R.1.2.1.1: Receipt of CREATE_CHILD_SA request ................................. 933

GROUP 2.2. USE OF RETRANSMISSION TIMERS .............................................................. 943
Test IKEv2.SGW.R.1.2.2.1: Receipt of CREATE_CHILD_SA requests................................. 943

GROUP 2.3. STATE SYNCHRONIZATION AND CONNECTION TIMEOUTS ............................. 945
Test IKEv2.SGW.R.1.2.3.1: Receiving Delete Payload for Multiple CHILD_SA................... 945

GROUP 2.4. CRYPTOGRAPHIC ALGORITHM NEGOTIATION.................................................. 949
Test IKEv2.SGW.R.1.2.4.1: Sending NO_PROPOSAL_CHOSEN ............................................ 949

GROUP 2.5. REKEYING CHILD_SA USING A CREATE_CHILD_SA EXCHANGE .................... 952
Test IKEv2.SGW.R.1.2.5.1: Close the replaced CHILD_SA ................................................. 952
Test IKEv2.SGW.R.1.2.5.2: Use of the new CHILD_SA ..................................................... 955
Test IKEv2.SGW.R.1.2.5.3: Receiving Multiple Transform .................................................. 958
Test IKEv2.SGW.R.1.2.5.4: Receiving Multiple Proposal ..................................................... 962
Test IKEv2.SGW.R.1.2.5.5: Perfect Forward Secrecy ......................................................... 966
Test IKEv2.SGW.R.1.2.5.6: Use of the old CHILD_SA ....................................................... 970

GROUP 2.6. REKEYING IKE_SAS USING A CREATE_CHILD_SA EXCHANGE ....................... 973
Test IKEv2.SGW.R.1.2.6.1: Sending CREATE_CHILD_SA response ..................................... 973
Test IKEv2.SGW.R.1.2.6.2: Receipt of cryptographically valid message on the old SA .......... 975
Test IKEv2.SGW.R.1.2.6.3: Receipt of cryptographically valid message on the new SA .......... 978
Test IKEv2.SGW.R.1.2.6.4: Close the replaced IKE_SA ...................................................... 981
Test IKEv2.SGW.R.1.2.6.5: Receiving Multiple Transform ................................................... 984
Test IKEv2.SGW.R.1.2.6.6: Receiving Multiple Proposal ...................................................... 988
Test IKEv2.SGW.R.1.2.6.7: Changing RPFs when rekeying the IKE_SA .............................. 993
Test IKEv2.SGW.R.1.2.6.8: D-H transform NONE when rekeying the IKE_SA .................... 996
Test IKEv2.SGW.R.1.2.6.9: Rekeying Failure ................................................................. 997

GROUP 2.7. CREATING NEW CHILD_SA WITH THE CREATE_CHILD_SA EXCHANGE ............... 1000
Test IKEv2.SGW.R.1.2.7.1: Receipt of cryptographically protected message on the new SA ...... 1000

GROUP 2.8. ERROR HANDLING....................................................................................... 1006
Test IKEv2.SGW.R.1.2.8.1: AUTHENTICATION_FAILED ................................................... 1006

GROUP 2.9. NON ZERO RESERVED FIELDS....................................................................... 1007
Test IKEv2.SGW.R.1.2.9.1: Non zero RESERVED fields in CREATE_CHILD_SA request ....... 1007

Group 3. The INFORMATIONAL Exchange.................................................................... 1009

GROUP 3.1. HEADER AND PAYLOAD FORMATS................................................................. 1009
Test IKEv2.SGW.R.1.3.1.1: Sending INFORMATIONAL response ........................................ 1009

GROUP 3.2. USE OF RETRANSMISSION TIMERS .............................................................. 1013
Test IKEv2.SGW.R.1.3.2.1: Receipt of retransmitted INFORMATIONAL request ................. 1013

GROUP 3.3. NON ZERO RESERVED FIELDS....................................................................... 1015
Test IKEv2.SGW.R.1.3.3.1: Non RESERVED fields in INFORMATIONAL request ............... 1015

Section 2.2.2. Endpoint to Security Gateway Tunnel......................................................... 1017

Group 1. The Initial Exchanges....................................................................................... 1039

GROUP 1.1. HEADER AND PAYLOAD FORMATS................................................................. 1040
Test IKEv2.SGW.R.2.1.1.1: Sending IKE_AUTH response ................................................... 1040
Test IKEv2.SGW.R.2.1.1.2: Use of CHILD_SA .................................................................. 1050

GROUP 1.2. REQUESTING AN INTERNAL ADDRESS ON A REMOTE NETWORK .................... 1052
Test IKEv2.SGW.R.2.1.2.1: Receipt of CFG_REQUEST ....................................................... 1052
Test IKEv2.SGW.R.2.1.2.2: Use of CHILD_SA ................................................................. 1056
Test IKEv2.SGW.R.2.1.2.3: Non zero RESERVED fields in Configuration Payload ................. 1060
Test IKEv2.SGW.R.2.1.2.4: No Configuration payload ......................................................... 1063
Test IKEv2.SGW.R.2.1.2.5: Receipt of Multiple CFG_REQUEST ........................................ 1065
Requirements
To obtain the IPv6 Ready Logo Phase-2 for IKEv2, the Node Under Test (NUT) must satisfy all of the following requirements.

Equipment Type
There are two possibilities for equipment types:

End-Node:
A node who can use IKEv2 (IPsec) only for itself. Host and Router can be an End-Node.

SGW (Security Gateway):
A node who can provide IKEv2 (IPsec tunnel mode) for nodes behind it. Router can be a SGW.

Function List

Basic/Advanced Functionality table
This conformance test specification consists following BASIC/ADVANCED functions. The tests for ADVANCED functions may be omitted if the NUT does not support the ADVANCED function. All NUTs are required to support BASIC. ADVANCED is required for all NUTs which support ADVANCED function.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>BASIC</th>
<th>ADVANCED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchange Type</td>
<td>Initial Exchanges (IKE_INIT, IKE_AUTH)</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>CREATE_CHILD_SA</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>INFORMATIONAL</td>
<td>-</td>
</tr>
<tr>
<td>IKE_SA</td>
<td>Encryption Algorithm ENCR_3DES</td>
<td>ENCR_AES_CBC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ENCR_AES_CTR</td>
</tr>
<tr>
<td></td>
<td>Pseudo-random Function PRF_HMAC_SHA1</td>
<td>PRF_AES128_XCBC</td>
</tr>
<tr>
<td></td>
<td>Integrity Algorithm AUTH_HMAC_SHA1_96</td>
<td>AUTH_AES_XCBC_96</td>
</tr>
<tr>
<td></td>
<td>Diffie-Hellman Group 2 (1024 MODP Group)</td>
<td>14 (2048-bit MODP Group)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>24 (2048-bit MODP Group with 256-bit Prime</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Order Subgroup)</td>
</tr>
<tr>
<td></td>
<td>Encryption Algorithm ENCR_3DES</td>
<td>ENCR_AES_CBC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ENCR_AES_CTR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ENCR_AES_CTR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ENCR_NULL</td>
</tr>
<tr>
<td></td>
<td>Integrity Algorithm AUTH_HMAC_SHA1_96</td>
<td>AUTH_AES_XCBC_96</td>
</tr>
<tr>
<td></td>
<td>Extended Sequence Numbers No Extended</td>
<td>Extended Sequence Numbers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Numbers</td>
</tr>
<tr>
<td>Authentication Method</td>
<td>PSK</td>
<td>-</td>
</tr>
<tr>
<td>Security Protocol</td>
<td>ESP</td>
<td>-</td>
</tr>
<tr>
<td>Encapsulation mode</td>
<td>End-Node Transport</td>
<td>Tunnel</td>
</tr>
<tr>
<td></td>
<td>SGW Tunnel</td>
<td>-</td>
</tr>
<tr>
<td>Multiple Proposals</td>
<td>Receiving</td>
<td>Sending</td>
</tr>
<tr>
<td>Multiple Transforms</td>
<td>Receiving</td>
<td>Sending</td>
</tr>
<tr>
<td>Liveness Check</td>
<td>Support</td>
<td>-</td>
</tr>
<tr>
<td>Feature</td>
<td>Support</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>Cookies</td>
<td>-</td>
<td>Support</td>
</tr>
<tr>
<td>Rekeying</td>
<td>Support</td>
<td>-</td>
</tr>
<tr>
<td>Traffic Selector Negotiation</td>
<td>-</td>
<td>Support</td>
</tr>
<tr>
<td>Requesting an Internal Address on a Remote Network</td>
<td>-</td>
<td>Support</td>
</tr>
<tr>
<td>Perfect Forward Secrecy</td>
<td>-</td>
<td>Support</td>
</tr>
<tr>
<td>Closing SAs</td>
<td>Support</td>
<td>-</td>
</tr>
<tr>
<td>ID Type</td>
<td>ID_IPV6_ADDR</td>
<td>-</td>
</tr>
<tr>
<td>Creating additional CHILD_SA</td>
<td>-</td>
<td>Support</td>
</tr>
</tbody>
</table>
Common Topology

Common Topology for End-Node: End-Node to End-Node

The common topology involves End-Nodes and Router device on each link.

The transport mode is used in this topology.
Common Topology for End-Node: End-Node to SGW

The common topology involves End-Node, SGW and Router device on each link.

The tunnel mode is used in this topology.
Common Topology for SGW: SGW to SGW

Prefix A = 2001:0db8:0001:0001::/64
Prefix B = 2001:0db8:0001:0002::/64
Prefix X = 2001:0db8:000f:0001::/64
Prefix Y = 2001:0db8:000f:0002::/64

The common topology involves SGWs, Router and Host device on each link.

The tunnel mode is used in this topology.
Common Topology for SGW: SGW to End-Node

Prefix A = 2001:0db8:0001:0001::/64
Prefix B = 2001:0db8:0001:0002::/64
Prefix X = 2001:0db8:000f:0001::/64

The common topology involves End-Node, SGW, Router and Host device on each link.
The tunnel mode is used in this topology.
Common Configuration for NUT

Common Configuration for End-Node: End-Node to End-Node

IKE Peer

<table>
<thead>
<tr>
<th>Address</th>
<th>Port</th>
<th>Method</th>
<th>Key Value</th>
<th>Type</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td>NUT</td>
<td>500</td>
<td>PSK</td>
<td>IKETEST12345678!</td>
<td>ID_IPV6_ADDR</td>
</tr>
<tr>
<td>Remote</td>
<td>TN1</td>
<td>500</td>
<td>PSK</td>
<td>IKETEST12345678!</td>
<td>ID_IPV6_ADDR</td>
</tr>
</tbody>
</table>

IKE_SA

<table>
<thead>
<tr>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>Diffie-Hellman</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>2 (1024 MODP Group)</td>
</tr>
</tbody>
</table>

If NUT is the initiator, above proposal must be one of proposals from NUT. If NUT is the responder, NUT must select above proposal.

CHILD_SA

<table>
<thead>
<tr>
<th>Security Protocol</th>
<th>Mode</th>
<th>Encryption</th>
<th>Integrity</th>
<th>Extended Sequence Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbound ESP</td>
<td>Transport</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>Outbound ESP</td>
<td>Transport</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No Extended Sequence Numbers</td>
</tr>
</tbody>
</table>

If NUT is the initiator, above proposal must be one of proposals from NUT. If NUT is the responder, NUT must select above proposal.

Traffic Selector

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Range</td>
<td>Address Range</td>
</tr>
<tr>
<td>Next Layer Protocol</td>
<td>Port Range</td>
</tr>
<tr>
<td>Port Protocol</td>
<td>Port Range</td>
</tr>
</tbody>
</table>

Inbound

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>TN1</td>
<td>ANY</td>
</tr>
<tr>
<td>ANY</td>
<td>NUT</td>
</tr>
<tr>
<td>ANY</td>
<td>ANY</td>
</tr>
</tbody>
</table>

Outbound

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUT</td>
<td>ANY</td>
</tr>
<tr>
<td>ANY</td>
<td>TN1</td>
</tr>
<tr>
<td>ANY</td>
<td>ANY</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 24 IPv6 Ready Logo Program IKEv2
Common Configuration for End-Node: End-Node to SGW

**IKE Peer**

<table>
<thead>
<tr>
<th>Address</th>
<th>Port</th>
<th>Authentication</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td>NUT</td>
<td>500 PSK</td>
<td>IKETEST123!</td>
</tr>
<tr>
<td>Remote</td>
<td>TN1 (Link X)</td>
<td>500 PSK</td>
<td>IKETEST456!</td>
</tr>
</tbody>
</table>

**IKE_SA**

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>Diffie-Hellman</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ENCR 3DES</td>
<td>PRF HMAC SHA1</td>
<td>AUTH HMAC SHA1 96</td>
<td>2 (1024 MODP Group)</td>
</tr>
</tbody>
</table>

If NUT is the initiator, above proposal must be one of proposals from NUT. If NUT is the responder, NUT must select above proposal.

**CHILD_SA**

<table>
<thead>
<tr>
<th>Security Protocol</th>
<th>Mode</th>
<th>Source</th>
<th>Encryption</th>
<th>Integrity</th>
<th>Extended Sequence Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbound ESP</td>
<td>Tunnel</td>
<td>Link Y</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>Outbound ESP</td>
<td>Tunnel</td>
<td>NUT</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
</tbody>
</table>

If NUT is the initiator, above proposal must be one of proposals from NUT. If NUT is the responder, NUT must select above proposal.

**Traffic Selector**

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Range</td>
<td>Next Layer Protocol</td>
</tr>
<tr>
<td>Inbound Link Y</td>
<td>ANY</td>
</tr>
<tr>
<td>Outbound NUT</td>
<td>ANY</td>
</tr>
</tbody>
</table>

If NUT is the initiator, NUT must propose Traffic Selector covering above address range. If NUT is the responder, NUT must narrow Traffic Selector to above address range.
Common Configuration for SGW: SGW to SGW

IKE Peer

<table>
<thead>
<tr>
<th>Address</th>
<th>Port</th>
<th>Authentication Method</th>
<th>Key Value</th>
<th>Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td>500</td>
<td>PSK</td>
<td>IKETEST123!</td>
<td>ID IPV6 ADDR</td>
<td>NUT (Link A)</td>
</tr>
<tr>
<td>Remote</td>
<td>500</td>
<td>PSK</td>
<td>IKETEST456!</td>
<td>ID IPV6 ADDR</td>
<td>TN1 (Link X)</td>
</tr>
</tbody>
</table>

IKE-SA

<table>
<thead>
<tr>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption</td>
</tr>
<tr>
<td>ENCR 3DES</td>
</tr>
</tbody>
</table>

If NUT is the initiator, above proposal must be one of proposals from NUT.
If NUT is the responder, NUT must select above proposal.

CHILD_SA

<table>
<thead>
<tr>
<th>Security Protocol</th>
<th>Mode</th>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption</td>
<td>Integrity</td>
<td>Extended Sequence Numbers</td>
</tr>
<tr>
<td>Inbound</td>
<td>ESP</td>
<td>Tunnel</td>
</tr>
<tr>
<td>Outbound</td>
<td>ESP</td>
<td>Tunnel</td>
</tr>
</tbody>
</table>

If NUT is the initiator, above proposal must be one of proposals from NUT.
If NUT is the responder, NUT must select above proposal.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
</tr>
<tr>
<td>Address Range</td>
</tr>
<tr>
<td>Inbound</td>
</tr>
<tr>
<td>Outbound</td>
</tr>
</tbody>
</table>

If NUT is the initiator, NUT must propose Traffic Selector covering above address range.
If NUT is the responder, NUT must narrow Traffic Selector to above address range.
Common Configuration for SGW: SGW to End-Node

IKE Peer

<table>
<thead>
<tr>
<th>Address</th>
<th>Port</th>
<th>Authentication</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td>500</td>
<td>PSK</td>
<td>IKETEST123!</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ID_IPV6_ADDR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NUT (Link A)</td>
</tr>
<tr>
<td>Remote</td>
<td>500</td>
<td>PSK</td>
<td>IKETEST456!</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ID_IPV6_ADDR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TN1</td>
</tr>
</tbody>
</table>

IKE_Sa

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>Diffie-Hellman</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ENCR_3DES</td>
<td>PRF HMAC SHA1</td>
<td>AUTH HMAC SHA1 96</td>
<td>2 (1024 MODP Group)</td>
</tr>
</tbody>
</table>

If NUT is the initiator, above proposal must be one of proposals from NUT.
If NUT is the responder, NUT must select above proposal.

CHILD_SA

<table>
<thead>
<tr>
<th>Security Protocol</th>
<th>Mode</th>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Encryption</td>
</tr>
<tr>
<td>Inbound</td>
<td>ESP</td>
<td>Tunnel</td>
</tr>
<tr>
<td>Outbound</td>
<td>ESP</td>
<td>Tunnel</td>
</tr>
</tbody>
</table>

If NUT is the initiator, above proposal must be one of proposals from NUT.
If NUT is the responder, NUT must select above proposal.

Traffic Selector

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Range</td>
<td>Next Layer Protocol</td>
</tr>
<tr>
<td>TN1</td>
<td>ANY</td>
</tr>
<tr>
<td>Outbound</td>
<td>Link B</td>
</tr>
</tbody>
</table>

If NUT is the initiator, NUT must propose Traffic Selector covering above address range.
If NUT is the responder, NUT must narrow Traffic Selector to above address range.
Common Packets
Common Packets to be transmitted from Tester are defined as the following tables. Tests in this test specification may refer to these common packets.

### IKE_SA_INIT Messages

**Common Packet #1: IKE_SA_INIT request**

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator's SPI</td>
<td>Any</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder's SPI</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>34 (IKE_SA_INIT)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>T (bit 3 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Next Payload</td>
<td>34 (KE)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td>KE Payload</td>
<td>Next Payload</td>
<td>40 (Ni, Nr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>136</td>
</tr>
<tr>
<td></td>
<td>DH Group #</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Key Exchange Data</td>
<td>any</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Nonce Data</td>
<td>any</td>
</tr>
</tbody>
</table>

- **SA Payload**

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>34 (KE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>44</td>
</tr>
<tr>
<td>Proposal #T</td>
<td>SA Proposal</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Proposal Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>1 (IKE)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td># of Transforms</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>SA Transform</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>2 (PRF)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1)</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>4 (D-H)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (1024 MODP Group)</td>
<td></td>
</tr>
</tbody>
</table>
## Common Packet #2: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE SA Initiator’s SPI</td>
<td>The same value as corresponding request’s IKE SA Initiator’s SPI value</td>
</tr>
<tr>
<td></td>
<td>IKE SA Responder’s SPI</td>
<td>Any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>34 (IKE_SA_INIT)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Next Payload</td>
<td>34 (KE)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>SA Proposals</td>
<td>See SA Table below</td>
</tr>
<tr>
<td>KE Payload</td>
<td>Next Payload</td>
<td>40 (Ni, Nr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>136</td>
</tr>
<tr>
<td></td>
<td>DH Group #</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Key Exchange Data</td>
<td>any</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Nonce Data</td>
<td>any</td>
</tr>
</tbody>
</table>

### SA Payload

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>34 (KE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>44</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Proposal Length</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Proposal #</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>1 (IKE)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td># of Transforms</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

#### SA Transform

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Protocol ID</th>
<th>3 (DES)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Transform ID</th>
<th>3 (DES)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>2 (PRF)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>2 (HMAC_SHA1)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Transform ID</th>
<th>2 (HMAC_SHA1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>2 (HMAC_SHA1)</td>
<td></td>
</tr>
</tbody>
</table>

---

IPv6 FORUM TECHNICAL DOCUMENT  30 IPv6 Ready Logo Program IKEv2
<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>8</td>
<td>4 (D-H)</td>
<td>0</td>
<td>2 (1024 MODP Group)</td>
</tr>
</tbody>
</table>
### IKE_AUTH Messages

**Common Packet #3: IKE_AUTH request for Transport Mode**

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP Header</th>
<th>Source Port</th>
<th>500</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IKEv2 Header</th>
<th>IKE SA Initiator's SPI</th>
<th>The IKE SA Initiator's SPI value used by this IKE message</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IKE SA Responder's SPI</td>
<td>The IKE SA Responder's SPI value used by this IKE message</td>
</tr>
<tr>
<td>Next Payload</td>
<td>46 (E)</td>
<td></td>
</tr>
<tr>
<td>Major Version</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Minor Version</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Exchange Type</td>
<td>35 (IKE_AUTH)</td>
<td></td>
</tr>
<tr>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>J (bit 3 of Flags)</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>V (bit 4 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>R (bit 5 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Message ID</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>any</td>
<td></td>
</tr>
</tbody>
</table>

**E Payload**

| Next Payload | 35 (IDi) |
| Critical | 0 |
| Reserved | 0 |
| Payload Length | any |
| Initialization Vector | The same value as block length of the underlying encryption algorithm |
| Encrypted IKE Payloads | Subsequent payloads encrypted by underlying encryption algorithm |
| Padding | Any value which to be a multiple of the encryption block size |
| Pad Length | The length of the Padding field |
| Integrity Checksum Data | The Cryptographic checksum of the entire message |

**IDi Payload**

| Next Payload | 39 (AUTH) |
| Critical | 0 |
| Reserved | 0 |
| Payload Length | 24 |
| ID Type | IPV6_ADDR |
| Reserved | 0 |
| Identification Data | TN1’s Global Address on Link X |

**AUTH Payload**

| Next Payload | 41 (N) |
| Critical | 0 |
| Reserved | 0 |
| Payload Length | any |
| Auth Method | 2 (SK_MIC) |
| Reserved | 0 |
| Authentication Data | any |

**N Payload**

| Next Payload | 33 (SA) |
| Critical | 0 |
| Reserved | 0 |
| Payload Length | 8 |
| Protocol ID | 0 |
| SPI Size | 0 |
| Notify Message Type | 16391 (USE_TRANSPORT_MODE) |

**SA Payload**

| Next Payload | 44 (TSi) |
| Critical | 0 |
| Reserved | 0 |
| Payload Length | 40 |

**TSi Payload**

| Next Payload | 45 (TSr) |
| Critical | 0 |
| Reserved | 0 |
| Payload Length | 48 |
| Number of TSs | 1 |

**TSr Payload**

| Next Payload | 0 |
| Critical | 0 |
| Reserved | 0 |
### SA Payload

<table>
<thead>
<tr>
<th>Payload Length</th>
<th>48</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of TSi</td>
<td>1</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See TSr Table below</td>
</tr>
</tbody>
</table>

- **SA Payload**

<table>
<thead>
<tr>
<th>Payload Length</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
</tbody>
</table>

- **TSi Payload for End-Node to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>

- **TSr Payload for End-Node to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
</tbody>
</table>
### Common Packet #4: IKE_AUTH response for Transport Mode

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>The same value as corresponding request’s IKE_SA Initiator’s SPI value</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>The same value as corresponding request’s IKE_SA Responder’s SPI value</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>35 (IKE_AUTH)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>36 (IDr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>Next Payload</td>
<td>39 (AUTH)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>ID Type</td>
<td>IPV6_ADDR</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Identification Data</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Auth Method</td>
<td>2 (SK_MIC)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Authentication Data</td>
<td>any</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type</td>
<td>16391 (USE_TRANSPORT_MODE)</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Next Payload</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Next Payload</td>
<td>45 (TSr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
</tbody>
</table>
SA Payload

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>Critical</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>44 (TSi)</td>
<td></td>
<td></td>
<td>40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>Critical</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td></td>
<td></td>
<td>36</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>Critical</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th># of Transforms</th>
<th>SPI Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SPI</th>
<th>any</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Critical</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>3 (more)</td>
<td></td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Transform</th>
<th>Transformer</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1 (ENCR)</td>
<td>8</td>
<td></td>
<td>3 (3DES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Transform</th>
<th>Transformer</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>3 (INTEG)</td>
<td>8</td>
<td></td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Transform</th>
<th>Transformer</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>5 (ESN)</td>
<td>8</td>
<td></td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

TSi Payload for End-Node to End-Node test cases

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic Selector</td>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT's Global Address on Link A</td>
<td>NUT's Global Address on Link A</td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT's Global Address on Link A</td>
<td>NUT's Global Address on Link A</td>
</tr>
</tbody>
</table>

TSr Payload for End-Node to End-Node test cases

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic Selector</td>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1's Global Address on Link X</td>
<td>TN1's Global Address on Link X</td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1's Global Address on Link X</td>
<td>TN1's Global Address on Link X</td>
</tr>
</tbody>
</table>
Common Packet #5: IKE_AUTH request for Tunnel Mode

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td></td>
</tr>
<tr>
<td>Source Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td></td>
</tr>
<tr>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td></td>
</tr>
<tr>
<td>IKE_SA Initiator's SPI</td>
<td>The IKE_SA Initiator’s SPI value used by this IKE message</td>
</tr>
<tr>
<td>IKE_SA Responder's SPI</td>
<td>The IKE_SA Responder’s SPI value used by this IKE message</td>
</tr>
<tr>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td>Exchange Type</td>
<td>35 (IKE_AUTH)</td>
</tr>
<tr>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>V (bit 3 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td>Message ID</td>
<td>1</td>
</tr>
<tr>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>35 (IDi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td>IDi Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>39 (AUTH)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>24</td>
</tr>
<tr>
<td>ID Type</td>
<td>IPv6_ADDR</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Identification Data</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td>Auth Method</td>
<td>2 (SK_MIC)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Authentication Data</td>
<td>any</td>
</tr>
<tr>
<td>SA Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
</tr>
<tr>
<td>TSi Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>45 (TSr)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See TSr Payload Table below</td>
</tr>
</tbody>
</table>

- **SA Payload**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>48</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 36 IPv6 Ready Logo Program IKEv2
### Proposal #1: SA Proposal

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>3</td>
</tr>
<tr>
<td>SPI</td>
<td>any</td>
</tr>
</tbody>
</table>

#### SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENC)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>5 (ESN)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

- **TSi Payload for End-Node to SGW test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y:0000:0000:0000:0000</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix Y::fff:fff:fff:fff</td>
<td></td>
</tr>
</tbody>
</table>

- **TSr Payload for End-Node to SGW test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
</tbody>
</table>

- **TSi Payload for SGW to SGW test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y:0000:0000:0000:0000</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix Y::fff:fff:fff:fff</td>
<td></td>
</tr>
</tbody>
</table>

- **TSr Payload for SGW to SGW test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
</tbody>
</table>
• TSi Payload for SGW to End-Node test cases

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8 (IPV6_ADDR_RANGE)</td>
<td>0 (any)</td>
<td>40</td>
<td>0</td>
<td>65535</td>
<td>Prefix B::0000:0000:0000:0000</td>
<td>Prefix B::ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>

• TSr Payload for SGW to End-Node test cases

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8 (IPV6_ADDR_RANGE)</td>
<td>0 (any)</td>
<td>40</td>
<td>0</td>
<td>65535</td>
<td>Prefix B::0000:0000:0000:0000</td>
<td>Prefix B::ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>
## Common Packet #6: IKE_AUTH response for Tunnel Mode

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>The same value as corresponding request’s IKE_SA Initiator’s SPI value</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>The same value as corresponding request’s IKE_SA Responder’s SPI value</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>35 (IKE_AUTH)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>36 (IDr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>Next Payload</td>
<td>39 (AUTH)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>ID Type</td>
<td>IPV6_ADDR</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Identification Data</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Auth Method</td>
<td>2 (SK_MIC)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Authentication Data</td>
<td>any</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Next Payload</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Next Payload</td>
<td>45 (TSr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Next Payload</td>
<td>48 (TSr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSr Payload Table below</td>
</tr>
</tbody>
</table>

- SA Payload

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>44 (TSi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 39 IPv6 Ready Logo Program IKEv2
### IPv6 Ready Logo Program IKEv2

**Proposal #1 SA Proposal**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>0</td>
</tr>
<tr>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>3</td>
</tr>
<tr>
<td>SPI</td>
<td>any</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload 3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload 3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload 0 (last)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>5 (ESN)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

### TSi Payload for End-Node to SGW test cases

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

### TSr Payload for End-Node to SGW test cases

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y:0000:0000:0000:0000:0000:0000:0000</td>
</tr>
</tbody>
</table>

### TSi Payload for SGW to SGW test cases

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix B:0000:0000:0000:0000</td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix B:ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>

### TSr Payload for SGW to SGW test cases

<table>
<thead>
<tr>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic Selector</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>IP Protocol ID</td>
</tr>
<tr>
<td>Selector Length</td>
</tr>
<tr>
<td>Start Port</td>
</tr>
<tr>
<td>End Port</td>
</tr>
<tr>
<td>Starting Address</td>
</tr>
<tr>
<td>Ending Address</td>
</tr>
</tbody>
</table>

- **TSi Payload for SGW to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix B::0000:0000:0000:0000</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix B::ffff:ffff:ffff:ffff</td>
<td></td>
</tr>
</tbody>
</table>

- **TSr Payload for SGW to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>
### CREATE_CHILD_SA Messages for Generating CHILD_SA

Common Packet #7: CREATE_CHILD_SA request for Generating CHILD_SA for Transport Mode

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>The IKE_SA Initiator’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>The IKE_SA Responder’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>36 (CREATE_CHILD_SA)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The value incremented the previous IKE message’s Message ID by one. If this message is first one, this value is set to 0.</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlining encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlining encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type</td>
<td>16391 (USE_TRANSPORT_MODE)</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Next Payload</td>
<td>40 (Ni, Nr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>SA Proposals</td>
<td>See SA Payload Table</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Next Payload</td>
<td>44 (TS)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Nonce Data</td>
<td>any</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Next Payload</td>
<td>45 (TSr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Next Payload</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSr Payload Table below</td>
</tr>
</tbody>
</table>

- SA Payload
<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>44 (TSi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Proposal #1</td>
<td>SA Proposal</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Proposal Length</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Proposal ID</td>
<td>3 (ESP)</td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td># of Transforms</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>SPI</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0 (last)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>5 (ESN)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>0 (No ESN)</td>
<td></td>
</tr>
</tbody>
</table>

- **TSi Payload for End-Node to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>

- **TSr Payload for End-Node to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
</tbody>
</table>
### Common Packet #8: CREATE_CHILD_SA response for Generating CHILD_SA for Transport Mode

**IPv6 Header**
- Source Address: TN1’s Global Address on Link X
- Destination Address: NUT’s Global Address on Link A

**UDP Header**
- Source Port: 500
- Destination Port: 500

**IKEv2 Header**
- **IKE_SA Initiator’s SPI**: The same value as corresponding request’s IKE_SA Initiator’s SPI value
- **IKE_SA Responder’s SPI**: The same value as corresponding request’s IKE_SA Responder’s SPI value
- Next Payload: 46 (E)
  - Major Version: 2
  - Minor Version: 0
  - Exchange Type: 36 (CREATE_CHILD_SA)
  - X (bits 0-2 of Flags): 0
  - I (bit 3 of Flags): any
  - V (bit 4 of Flags): 0
  - R (bit 5 of Flags): 1
  - X (bits 6-7 Flags): 0
  - Message ID: The same value as corresponding request's Message ID
  - Length: any

**E Payload**
- Next Payload: 41 (N)
  - Critical: 0
  - Reserved: 0
  - Payload Length: any
  - Initialization Vector: The same value as block length of the underlying encryption algorithm
  - Encrypted IKE Payloads: Subsequent payloads encrypted by underlying encryption algorithm
  - Padding: Any value which to be a multiple of the encryption block size
  - Pad Length: The length of the Padding field

**N Payload**
- Next Payload: 33 (SA)
  - Critical: 0
  - Reserved: 0
  - Payload Length: 8
  - Protocol ID: 0
  - SPI Size: 0

**SA Payload**
- Notify Message Type: 16391 (USE_TRANSPORT_MODE)
- Next Payload: 40 (Ni, Nr)
  - Critical: 0
  - Payload Length: 40

**Ni, Nr Payload**
- Next Payload: 44 (TSi)
  - Critical: 0
  - Payload Length: any

**TSi Payload**
- Next Payload: 45 (TSr)
  - Critical: 0
  - Payload Length: 48

**TSr Payload**
- Next Payload: 44 (TSr)
  - Critical: 0
  - Payload Length: 48

- **SA Payload**
  - **Next Payload**: 44 (TSi)
    - Critical: 0
    - Reserved: 0

---

**IPv6 FORUM TECHNICAL DOCUMENT**

IPv6 Ready Logo Program IKEv2
<table>
<thead>
<tr>
<th>Payload Length</th>
<th>SA Proposal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal #1</td>
<td>Next Payload 40 (last)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>3</td>
</tr>
<tr>
<td>SPI</td>
<td>any</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload 3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENC)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload 3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload 0 (last)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>5 (ESN)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

- **TSi Payload for End-Node to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>TS Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
<td></td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
</tbody>
</table>

- **TSr Payload for End-Node to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>TS Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
<td></td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>
Common Packet #9: CREATE_CHILD_SA request for Generating CHILD_SA for Tunnel Mode

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE SA Initiator’s SPI</td>
<td>The IKE SA Initiator’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>IKE SA Responder’s SPI</td>
<td>The IKE SA Responder’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>36 (CREATE_CHILD_SA)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The value incremented the previous IKE message’s Message ID by one. If this message is first one, this value is set to 0.</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Next Payload</td>
<td>40 (Ni, Nr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Nonce Data</td>
<td>any</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Next Payload</td>
<td>45 (TSi)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Next Payload</td>
<td>48 (TSr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of TSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSr Payload Table below</td>
</tr>
</tbody>
</table>

- **SA Payload**

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>44 (TSi)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td>Proposal #1</td>
<td>SA Proposal</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Proposal</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
</tbody>
</table>
### SPI Size
- 4

### # of Transforms
- 3

### SPI
- any

### SA Transform
- **Next Payload**: 3 (more)
- **Reserved**: 0
- **Transform Length**: 8
- **Transform Type**: 1 (ENCRC)
- **Reserved**: 0
- **Transform ID**: 3 (3DES)

### SA Transform
- **Next Payload**: 3 (more)
- **Reserved**: 0
- **Transform Length**: 8
- **Transform Type**: 3 (INTEG)
- **Reserved**: 0
- **Transform ID**: 2 (HMAC_SHA1_96)

### SA Transform
- **Next Payload**: 0 (last)
- **Reserved**: 0
- **Transform Length**: 8
- **Transform Type**: 5 (ESN)
- **Reserved**: 0
- **Transform ID**: 0 (No ESN)

---

### TSı Payload for SGW to SGW test cases

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8 (IPV6_ADDR_RANGE)</td>
<td>0 (any)</td>
<td>40</td>
<td></td>
<td>65535</td>
<td>Prefix Y:0000:0000:0000:0000:0000</td>
<td></td>
</tr>
</tbody>
</table>

### TSr Payload for SGW to SGW test cases

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8 (IPV6_ADDR_RANGE)</td>
<td>0 (any)</td>
<td>40</td>
<td></td>
<td>65535</td>
<td>Prefix B:0000:0000:0000:0000:0000</td>
<td></td>
</tr>
</tbody>
</table>
Common Packet #10: CREATE_CHILD_SA response for Generating CHILD_SA for Tunnel Mode

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>The same value as corresponding request’s IKE_SA Initiator’s SPI value</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>The same value as corresponding request’s IKE_SA Responder’s SPI value</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>36 (CREATE_CHILD_SA)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request’s Message ID</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
</tbody>
</table>

| E Payload | Next Payload | 33 (SA) |
|           | Critical | 0 |
|           | Reserved | 0 |
|           | Payload Length | any |
|           | Initialization Vector | any |
|           | Encrypted IKE Payloads | any |
|           | Padding | any |
|           | Pad Length | any |
|           | Integrity Checksum Data | any |

| SA Payload | Next Payload | 40 (Ni, Nr) |
|           | Critical | 0 |
|           | Reserved | 0 |
|           | Payload Length | 0 |
|           | SA Proposals | See SA Payload Table below |

| Ni, Nr Payload | Next Payload | 44 (TSi) |
|               | Critical | 0 |
|               | Reserved | 0 |
|               | Payload Length | 0 |
|               | Nonce Data | 40 |

| TSi Payload | Next Payload | 45 (TSi) |
|            | Critical | 0 |
|            | Reserved | 0 |
|            | Payload Length | 48 |
|            | Number of TSs | 1 |
|            | Reserved | 0 |
|            | Traffic Selectors | See TSi Payload Table below |

| TSr Payload | Next Payload | 48 |
|            | Critical | 0 |
|            | Reserved | 0 |
|            | Payload Length | 48 |
|            | Number of TSs | 1 |
|            | Reserved | 0 |
|            | Traffic Selectors | See TSr Payload Table below |

- **SA Payload**

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>44 (TSi)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td>Proposal #1</td>
<td>SA Proposal</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>SPI</td>
<td>any</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>5 (ESN)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

- **TSi Payload for SGW to SGW test cases**

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Starting Address</td>
<td>B:0000:0000:0000:0000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ending Address</td>
<td>B:ffff:ffff:ffff:ffff</td>
<td></td>
</tr>
</tbody>
</table>

- **TSr Payload for SGW to SGW test cases**

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Starting Address</td>
<td>Y:0000:0000:0000:0000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ending Address</td>
<td>Y:ffff:ffff:ffff:ffff</td>
<td></td>
</tr>
</tbody>
</table>
CREATE_CHILD_SA Messages for Rekeying IKE_SA

Common Packet #11: CREATE_CHILD_SA request for Rekeying IKE_SA

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>The IKE_SA Initiator’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>The IKE_SA Responder’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>36 (CREATE_CHILD_SA)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The value incremented the previous IKE message’s Message ID by one. If this message is first one, this value is set to 0.</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>E Payload Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
<tr>
<td></td>
<td>SA Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>44</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Nonce Data</td>
<td>any</td>
</tr>
</tbody>
</table>

- **SA Payload**

| SA Payload | Next Payload | 34 (KE) |
|            | Critical | 0 |
|            | Reserved | 0 |
|            | Payload Length | 44 |
| Proposal #1 | SA Proposal | Next Payload | 0 (last) |
|            | Reserved | 0 |
|            | Proposal Length | 40 |
|            | Proposal # | 1 |
|            | Protocol ID | 1 (IKE) |
|            | SPI Size | 0 |
| # of Transforms | 4 |
| SA Transform | Next Payload | 3 (more) |
|                | Reserved | 0 |
|                | Transform Length | 8 |
|                | Transform Type | 1 (ENCR) |
|                | Reserved | 0 |
|                | Transform ID | 3 (3DES) |
|                | SA Transform | Next Payload | 3 (more) |
|                | Reserved | 0 |
|                | Transform Length | 8 |
|                | Transform Type | 2 (PRF) |
| SA Transform | Reserved | 0 |
| SA Transform | Transform ID | 2 (HMAC_SHA1) |
| SA Transform | Next Payload | 3 (more) |
| SA Transform | Reserved | 0 |
| SA Transform | Transform Length | 8 |
| SA Transform | Transform Type | 3 (INTEG) |
| SA Transform | Reserved | 0 |
| SA Transform | Transform ID | 2 (HMAC_SHA1_96) |
| SA Transform | Next Payload | 0 (last) |
| SA Transform | Reserved | 0 |
| SA Transform | Transform Length | 8 |
| SA Transform | Transform Type | 4 (D-H) |
| SA Transform | Reserved | 0 |
| SA Transform | Transform ID | 2 (1024 MODP Group) |
### Common Packet #12: CREATE_CHILD_SA response for Rekeying IKE_SA

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP Header</th>
<th>Source Port</th>
<th>500</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
</tbody>
</table>

| IKEv2 Header | IKE_SA Initiator’s SPI | The same value as corresponding request’s IKE_SA Initiator’s SPI value |
|             | IKE_SA Responder’s SPI | The same value as corresponding request’s IKE_SA Responder’s SPI value |
|             | Next Payload | 46 (E) |
|             | Major Version | 2 |
|             | Minor Version | 0 |
|             | Exchange Type | 36 (CREATE_CHILD_SA) |
|             | X (bits 0-2 of Flags) | 0 |
|             | I (bit 3 of Flags) | any |
|             | V (bit 4 of Flags) | 0 |
|             | R (bit 5 of Flags) | 1 |
|             | X (bits 6-7 Flags) | 0 |
|             | Message ID Length | The same value as corresponding request's Message ID Length |

<table>
<thead>
<tr>
<th>E Payload</th>
<th>Next Payload</th>
<th>33 (SA)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding Pad Length</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>40 (Ni, Nr)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ni, Nr Payload</th>
<th>Next Payload</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Nonce Data</td>
<td>any</td>
</tr>
</tbody>
</table>

- **SA Payload**

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>34 (KE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>44</td>
</tr>
<tr>
<td>Proposal #1</td>
<td>SA Proposal</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal Length</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protocol ID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
</tr>
<tr>
<td></td>
<td></td>
<td># of Transforms</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>2 (PRF)</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>2 (HMAC_SHA1)</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 52 IPv6 Ready Logo Program IKEv2
<table>
<thead>
<tr>
<th></th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA Transform</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>2 (HMAC_SHA1_96)</td>
<td>0 (last)</td>
<td>0</td>
<td>4 (D-H)</td>
<td>0</td>
<td>2 (1024 MODP Group)</td>
</tr>
</tbody>
</table>
CREATE_CHILD_SA Messages for Rekeying CHILD_SA

Common Packet #13: CREATE_CHILD_SA request for Rekeying CHILD_SA for Transport Mode

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Address</td>
<td>TNT’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>UDP Header</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Port</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>Destination Port</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IKE_SA Initiator's SPI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IKE_SA Responder's SPI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>46 (E)</td>
<td></td>
</tr>
<tr>
<td>Major Version</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Minor Version</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Exchange Type</td>
<td>36 (CREATE_CHILD_SA)</td>
<td></td>
</tr>
<tr>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>I (bit 3 of Flags)</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>V (bit 4 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>R (bit 5 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Message ID</td>
<td>The value incremented the previous IKE message’s Message ID by one. If this message is first one, this value is set to 0.</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
<td></td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
<td></td>
</tr>
<tr>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>16393 (REKEY_SA)</td>
<td></td>
</tr>
<tr>
<td>SPI</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>33 (SA)</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>16391 (USE_TRANSPORT_MODE)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>40 (Ni, Nr)</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>44 (TSi)</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Nonce Data</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSi Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>45 (TSr)</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>Number of TSs</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSr Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
### SA Payload

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>48</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**TSi Payload for End-Node to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>

**TSr Payload for End-Node to End-Node test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
</tbody>
</table>
Common Packet #14: CREATE_CHILD_SA response for Rekeying CHILD_SA for Transport Mode

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator's SPI</td>
<td>The same value as corresponding request’s IKE_SA Initiator’s SPI value</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder's SPI</td>
<td>The same value as corresponding request’s IKE_SA Responder’s SPI value</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request's Message ID</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type</td>
<td>16391 (USE_TRANSPORT_MODE)</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Next Payload</td>
<td>40 (Ni, Nr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Next Payload</td>
<td>45 (TSi)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Nonce Data</td>
<td>any</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Next Payload</td>
<td>48 (TSr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Number of TSSs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSr Payload Table below</td>
</tr>
</tbody>
</table>

- **SA Payload**

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>44 (TSi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>Number of TSSs</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
<td></td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See TSr Payload Table below</td>
<td></td>
</tr>
</tbody>
</table>
### IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Payload Length</th>
<th>Proposal #1 SA Proposal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Next Payload 0 (last)</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Proposal Length 36</td>
</tr>
<tr>
<td></td>
<td>Proposal # 1</td>
</tr>
<tr>
<td></td>
<td>Proposal ID 3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size 4</td>
</tr>
<tr>
<td></td>
<td># of Transforms 3</td>
</tr>
<tr>
<td></td>
<td>SPI any</td>
</tr>
<tr>
<td></td>
<td>SA Transform Next Payload 3 (more)</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Transform Length 8</td>
</tr>
<tr>
<td></td>
<td>Transform Type 1 (ENCR)</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Transform ID 3 (3DES)</td>
</tr>
<tr>
<td></td>
<td>SA Transform Next Payload 3 (more)</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Transform Length 8</td>
</tr>
<tr>
<td></td>
<td>Transform Type 3 (INTEG)</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Transform ID 2 (HMAC_SHA1_96)</td>
</tr>
<tr>
<td></td>
<td>SA Transform Next Payload 0 (last)</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Transform Length 8</td>
</tr>
<tr>
<td></td>
<td>Transform Type 5 (ESN)</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Transform ID 0 (No ESN)</td>
</tr>
</tbody>
</table>

#### TSi Payload for End-Node to End-Node test cases

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

#### TSr Payload for End-Node to End-Node test cases

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
</tbody>
</table>
Common Packet #15: CREATE_CHILD_SA request for Rekeying CHILD_SA for Tunnel Mode

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>The IKE-SA Initiator’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>The IKE-SA Responder’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>36 (CREATE_CHILD_SA)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The value incremented the previous IKE message’s Message ID by one. If this message is first one, this value is set to 0.</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type</td>
<td>16393 (REKEY_SA)</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Next Payload</td>
<td>40 (Ni, Nr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>SA Proposals</td>
<td>See SA Payload Table below</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Next Payload</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>_nonce Data</td>
<td>any</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Next Payload</td>
<td>45 (TSr)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of Tsxs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSi Payload Table below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>Number of Tsxs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See TSr Payload Table below</td>
</tr>
</tbody>
</table>

- SA Payload
### IPv6 Ready Logo Program IKEv2

#### Proposal #1

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>3</td>
</tr>
</tbody>
</table>

#### SA Transform

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
</tbody>
</table>

#### Proposal #1 SA Proposal

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
</tbody>
</table>

### TSi Payload for SGW to SGW test cases

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IPv6_ADDR_RANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y:ffff:ffff:ffff:ffff</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix Y:ffff:ffff:ffff:ffff</td>
<td></td>
</tr>
</tbody>
</table>

### TSr Payload for SGW to SGW test cases

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IPv6_ADDR_RANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix B:0000:0000:0000:0000:0000:0000:0000:000</td>
<td></td>
</tr>
</tbody>
</table>
Common Packet #16: CREATE_CHILD_SA response for Rekeying CHILD_SA for Tunnel Mode

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator's SPI</td>
<td>The same value as corresponding request’s IKE_SA Initiator’s SPI value</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder's SPI</td>
<td>The same value as corresponding request’s IKE_SA Responder’s SPI value</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>36 (CREATE_CHILD_SA)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request's Message ID</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
</tbody>
</table>

| SA Payload | Next Payload | 40 (Ni, Nr) |
|            | Critical | 0 |
|            | Reserved | 0 |
|            | Payload Length | 40 |
| Ni, Nr Payload | Next Payload | 44 (TSi) |
|            | Critical | 0 |
|            | Payload Length | any |
|            | Nonce Data | any |
| TSi Payload | Next Payload | 45 (TSi) |
|            | Critical | 0 |
|            | Payload Length | 48 |
|            | Number of TSs | 1 |
|            | Reserved | 0 |
|            | Traffic Selectors | See TSi Payload Table below |
| TSr Payload | Next Payload | 0 |
|            | Critical | 0 |
|            | Reserved | 0 |
|            | Payload Length | 48 |
|            | Number of TSs | 1 |
|            | Reserved | 0 |
|            | Traffic Selectors | See TSr Payload Table below |

- **SA Payload**

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>44 (TSi)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>40</td>
</tr>
<tr>
<td>Proposal #1</td>
<td>SA Proposal</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

# of Transforms | 3
---|---
SPI | any

### SA Transform

| Next Payload | 3 (more) |
| Reserved | 0 |
| Transform Length | 8 |
| Transform Type | 1 (ENCR) |
| Reserved | 0 |
| Transform ID | 3 (3DES) |

### SA Transform

| Next Payload | 3 (more) |
| Reserved | 0 |
| Transform Length | 8 |
| Transform Type | 3 (INTEG) |
| Reserved | 0 |
| Transform ID | 2 (HMAC_SHA1_96) |

### SA Transform

| Next Payload | 0 (last) |
| Reserved | 0 |
| Transform Length | 8 |
| Transform Type | 5 (ESN) |
| Reserved | 0 |
| Transform ID | 0 (No ESN) |

- **TSi Payload for SGW to SGW test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix B:0000:0000:0000:0000</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix B:ffff:ffff:ffff:ffff</td>
<td></td>
</tr>
</tbody>
</table>

- **TSr Payload for SGW to SGW test cases**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y:0000:0000:0000:0000</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix Y:ffff:ffff:ffff:ffff</td>
<td></td>
</tr>
</tbody>
</table>
### INFORMATIONAL Messages

**Common Packet #17: INFORMATIONAL request**

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP Header</th>
<th>Source Port</th>
<th>300</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IKEv2 Header</th>
<th>IKE SA Initiator's SPI</th>
<th>The IKE SA Initiator’s SPI value used by this IKE message</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IKE SA Responder's SPI</td>
<td>The IKE SA Responder’s SPI value used by this IKE message</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The value incremented the previous IKE message’s Message ID by one. If this message is first one, this value is set to 0.</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pld Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
</tbody>
</table>
### Common Packet #18: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TNU’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE-SA Initiator's SPI</td>
<td>The same value as corresponding request’s IKE-SA Initiator’s SPI value</td>
</tr>
<tr>
<td></td>
<td>IKE-SA Responder's SPI</td>
<td>The same value as corresponding request’s IKE-SA Responder’s SPI value</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request’s Message ID</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The cryptographic checksum of the entire message</td>
</tr>
</tbody>
</table>
### ICMPv6 Echo Requests

#### Common Packet #19: ICMPv6 Echo Request for End-Node to End-Node test cases

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
</tr>
<tr>
<td></td>
<td>TN1's Global Address</td>
</tr>
<tr>
<td>Destination Address</td>
<td>NUT's Global Address</td>
</tr>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
</tr>
<tr>
<td></td>
<td>CHILD_SA’s SPI value used by this message</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
</tr>
<tr>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td>Next Header</td>
<td>58 (IPv6-ICMP)</td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
</tr>
</tbody>
</table>

#### Common Packet #20: ICMPv6 Echo Request for End-Node to SGW test cases

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
</tr>
<tr>
<td></td>
<td>TN1's Global Address on Link X</td>
</tr>
<tr>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
</tr>
<tr>
<td></td>
<td>CHILD_SA’s SPI value used by this message</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
</tr>
<tr>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td>Next Header</td>
<td>41 (IPv6)</td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
</tr>
</tbody>
</table>

#### Common Packet #21: ICMPv6 Echo Request for SGW to SGW test cases

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
</tr>
<tr>
<td></td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
</tr>
<tr>
<td></td>
<td>CHILD_SA’s SPI value used by this message</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
</tr>
<tr>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td>Next Header</td>
<td>41 (IPv6)</td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
</tr>
</tbody>
</table>

#### Common Packet #22: ICMPv6 Echo Request for SGW to End-Node test cases

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
</tr>
<tr>
<td></td>
<td>TN1’s Global Address</td>
</tr>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
</tr>
<tr>
<td></td>
<td>CHILD_SA’s SPI value used by this message</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
</tr>
<tr>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td>Next Header</td>
<td>41 (IPv6)</td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
</tr>
<tr>
<td>ESP</td>
<td>Destination Address</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Sequence Number</td>
<td></td>
</tr>
<tr>
<td>Payload Data</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td></td>
</tr>
<tr>
<td>Pad Length</td>
<td></td>
</tr>
<tr>
<td>Next Header</td>
<td></td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td></td>
</tr>
</tbody>
</table>

| IPv6 Header                  |                     |                                |                          |                                           |
| Source Address               | TNT’s Global Address |                                |                          |                                           |
| Destination Address          | TH1’s Global Address |                                |                          |                                           |

| ICMPv6 Header                |                     |                                |                          |                                           |
| Type                         | 128                 |                                |                          |                                           |
| Code                         | 0                   |                                |                          |                                           |
| Identifier                   | any                 |                                |                          |                                           |
| Sequence Number              | any                 |                                |                          |                                           |
| Payload Data                 | 0x0000000000000000  |                                |                          |                                           |
ICMPv6 Echo Replies

Common Packet #23: ICMPv6 Echo Reply for End-Node to End-Node test cases

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address</td>
</tr>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
<td>CHILD SA’s SPI value used by this message</td>
</tr>
<tr>
<td></td>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
</tr>
<tr>
<td></td>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Next Header</td>
<td>58 (IPv6-ICMP)</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td>Code</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Identifier</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Sequence Number</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Payload Data</td>
<td>0x0000000000000000</td>
</tr>
</tbody>
</table>

Common Packet #24: ICMPv6 Echo Reply for End-Node to SGW test cases

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>NUT’s Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
<td>CHILD SA’s SPI value used by this message</td>
</tr>
<tr>
<td></td>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
</tr>
<tr>
<td></td>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Next Header</td>
<td>41 (IPv6)</td>
</tr>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
<td>TH1’s Global Address</td>
</tr>
<tr>
<td></td>
<td>Destination Address</td>
<td>TH1’s Global Address</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td>Code</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Identifier</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Sequence Number</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Payload Data</td>
<td>0x0000000000000000</td>
</tr>
</tbody>
</table>

Common Packet #25: ICMPv6 Echo Reply for SGW to SGW test cases

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TH1’s Global Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>TH2’s Global Address</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td>Code</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Identifier</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Sequence Number</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Payload Data</td>
<td>0x0000000000000000</td>
</tr>
</tbody>
</table>

Common Packet #26: ICMPv6 Echo Reply for SGW to End-Node test cases

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TH1’s Global Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>TN1’s Global Address</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td>Code</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Identifier</td>
<td>Any</td>
</tr>
<tr>
<td></td>
<td>Sequence Number</td>
<td>Any</td>
</tr>
<tr>
<td></td>
<td>Payload Data</td>
<td>0x0000000000000000</td>
</tr>
</tbody>
</table>
Section 1. End Node
Section 1.1. Initiator
Section 1.1.1. Endpoint-to-Endpoint Transport
Group 1. The Initial Exchanges
Group 1.1. Header and Payload Formats

Test IKEv2.EN.I.1.1.1.1: Sending IKE_SA_INIT request

Purpose:

To verify an IKEv2 device transmits IKE_SA_INIT request using properly Header and Payloads format.

References:

- [RFC4306] - Section 1.2, 2.10, 3.1, 3.2, 3.3, 3.4 and 3.9
- [RFC 4718] - Sections 7.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: IKE Header Format (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Part B: SA Payload Format (BASIC)
3. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
4. Observe the messages transmitted on Link A.

Part C: KE Payload Format (BASIC)
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.

Part D: Nonce Payload Format (BASIC)
7. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including properly formatted IKE Header containing following values:

- An IKE_SA Initiator’s SPI field set to a 64-bits value chosen by the NUT. It MUST not be zero.
- An IKE_SA Responder’s SPI field set to zero.
- A Next Payload field set to SA Payload (33).
- A Major Version field is set to 2.
- A Minor Version field is set to zero.
- An Exchange Type field is set to IKE_SA_INIT (34).
- A Flags field is set to (00010000)2 = (16)10.
- A Message ID field is set to zero.
- A Length field is set to the length of the message (header + payloads) in octets.

**Part B**

**Step 4: Judgment #1**
The NUT transmits an IKE_SA_INIT request including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field is set to KE Payload (34).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.

The following proposal must be included in Proposals field:

- A 0 or 2 field is set to zero if this structure is the last proposal, otherwise set to 2.
- A RESREVD field is set to zero.
- A Proposal Length field is set to length of this proposal, including all transforms and attributes. It is 40 bytes for this proposal according to Common Configuration.
- A Proposal # field is set to 1 if this structure is the first proposal, otherwise set to 1 greater than the previous proposal.
- A Protocol ID field is set to IKE (1).
- A SPI Size field is set to zero.
- A # of Transforms field is set to 4.

A Transform field is set to following (There are 4 Transform Structures).
Figure 5 Transform sub-structure format

Transform #1

- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field is set to ENCR (1).
- A RESERVED field is set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2

- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for PRF_HMAC_SHA1.
- A Transform Type field is set to PRF (2).
- A RESERVED field is set to zero.
- A Transform ID set to PRF_HMAC_SHA1 (2).

Transform #3

- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field is set to INTEG (3).
- A RESERVED field is set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #4

- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for 1024 MODP Group.
- A Transform Type field is set to D-H (4).
- A RESERVED field is set to zero.
- A Transform ID set to Group2 (2).
**Part C**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including properly formatted KE Payload containing following values:

- A Next Payload field is set to Nonce Payload (40).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 136 bytes for Group 2.
- A DH Group field is set to Group2 (2).
- A RESERVED field is set to zero.
- A Key Exchange Data field is set to Diffie-Hellman public value. The length of the Key Exchange Data field must be equal to 1024 bit.

**Figure 6 KE Payload format**

### Possible Problems:

- IKE_SA_INIT request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

---

**Part D**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT request including properly formatted Nonce Payload containing following values:

- A Next Payload field is set to zero.
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Nonce Data field is set to random data generated by the transmitting entity. The size of the Nonce must between 16 and 256 octets.

**Figure 7 Nonce Payload format**

### Possible Problems:

- IKE_SA_INIT request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.
- The implementation may not set single proposal by the implementation policy. In this case, Security Association Payload contains multiple proposals.

- Each of transforms can be located in the any order.
Test IKEv2.EN.I.1.1.1.2: Sending IKE_AUTH request

Purpose:

To verify an IKEv2 device transmits IKE_AUTH request using properly Header and Payloads format.

References:

- [RFC 4306] - Sections 1.2, 2.15, 3.1, 3.2, 3.3, 3.5, 3.8, 3.10, 3.13 and 3.14

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: IKE Header Format (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.

Part B: Encrypted Payload Format (BASIC)
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_SA_INIT response to the NUT.
8. Observe the messages transmitted on Link A.

Part C: IDi Payload Format (BASIC)
9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. TN1 responds with an IKE_SA_INIT response to the NUT.
12. Observe the messages transmitted on Link A.

**Part D: AUTH Payload Format (BASIC)**
13. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. TN1 responds with an IKE_SA_INIT response to the NUT.
16. Observe the messages transmitted on Link A.

**Part E: Notify Payload Format (BASIC)**
17. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. TN1 responds with an IKE_SA_INIT response to the NUT.
20. Observe the messages transmitted on Link A.

**Part F: SA Payload Format (BASIC)**
21. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
22. Observe the messages transmitted on Link A.
23. TN1 responds with an IKE_SA_INIT response to the NUT.
24. Observe the messages transmitted on Link A.

**Part G: TSi Payload Format (BASIC)**
25. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. TN1 responds with an IKE_SA_INIT response to the NUT.
28. Observe the messages transmitted on Link A.

**Part H: TSr Payload Format (BASIC)**
29. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
30. Observe the messages transmitted on Link A.
31. TN1 responds with an IKE_SA_INIT response to the NUT.
32. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted IKE Header containing following values:
An IKE_SA Initiator’s SPI field is set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.

An IKE_SA Responder’s SPI field is set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.

A Next Payload field is set to Encrypted Payload (46).

A Major Version field is set to 2.

A Minor Version field is set to zero.

An Exchange Type field is set to IKE_AUTH (35).

A Flags field is set to 0x10 (16).

A Message ID field is set to 1.

A Length field is set to the length of the message (header + payloads) in octets.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted Encrypted Payload containing following values:
- A Next Payload field is set to ID_IPV6_ADDR (35).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Check sum Data.
- An Initialization Vector field is set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
- An Encrypted IKE Payloads field is set to subsequent payloads encrypted by ENCR_3DES.
- A Padding field is set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
- A Pad Length field is set to the length of the Padding field.
- An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted ID Payload containing following values:

![Figure 10 ID Payload format]

- A Next Payload field is set to AUTH Payload (39).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 24 bytes for ID_IPV6_ADDR.
- An ID Type field is set to ID_IPV6_ADDR (5).
- A RESERVED field is set to zero.
- An Identification Data field is set to the NUT address.

**Part D**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted AUTH Payload containing following values:

![Figure 11 AUTH Payload format](image)

- A Next Payload field is set to Notify Payload (41).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 28 bytes for PRF_HMAC_SHA1.
- An Auth Method field is set to Shared Key Message Integrity Code (2).
- A RESERVED field is set to zero.
- An Authentication Data field is set to correct authentication value according to the manner described in RFC. It is 160 bytes length in PRF_HMAC_SHA1 case.

**Part E**

**Step 18: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted Notify Payload containing following values:
Figure 12 Notify Payload format

- A Next Payload field is set to SA Payload (33).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 8 bytes for USE_TRANSPORT_MODE.
- A Protocol ID field is set to undefined (0).
- A SPI Size field is set to zero.
- A Notify Message Type field is set to USE_TRANSPORT_MODE (16391)

**Part F**

**Step 22: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 24: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted SA Payload containing following values (refer following figures):

**Figure 14 SA Payload format**

- A Next Payload field is set to TSi Payload (44).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.

The following proposal must be included in Proposals field.

**Figure 15 Proposal sub-structure format**

Proposal #1

- A 0 or 2 field is set to zero if this structure is the last proposal, otherwise set to 2.
- A RESREVD field is set to zero.
- A Proposal Length field is set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field is set to 1 if this structure is the first proposal, otherwise set to 1 greater than the previous proposal.
- A Protocol ID field is set to ESP (3).
- A SPI Size field is set to 4.
- A # of Transforms field is set to 3.
- A SPI field is set to the sending entity’s SPI (4 octets value)

Transform field is set to following (There are 3 Transform Structures).
Transform #1

- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field is set to ENCR (1).
- A RESERVED field is set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2

- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field is set to INTEG (3).
- A RESERVED field is set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3

- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
- A Transform Type field is set to ESN (5).
- A RESERVED field is set to zero.
- A Transform ID set to No Extended Sequence Numbers (0).

Part G

Step 26: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 28: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted TSi Payload containing following values:
Figure 17 TSi Payload format

- A Next Payload field is set to TSr Payload (45).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Number of TSs field is set to the number of actual traffic selectors.
- A RESERVED field is set to zero.

The following traffic selector must be included in Traffic Selectors field.

Figure 18 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to less than or equal to NUT address.
- A Ending Address field is set to greater than or equal to NUT address.

Part H

Step 30: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 32: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted TSr Payload containing following values:

- A Next Payload field is set to zero.
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Number of TSs field is set to the number of actual traffic selectors.
- A RESERVED field is set to zero.

The following traffic selector must be included in Traffic Selectors field.

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to less than or equal to TN1 address.
- An Ending Address field is set to less than or equal to TN1 address.

Possible Problems:

- IKE_AUTH request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload...
may be different from this sample.

- The implementation may not set single proposal by the implementation policy. In this case, Security Association Payload contains multiple proposals.

- Each of transforms can be located in the any order.

- The implementation may not set single traffic selector by the implementation policy. In this case, Traffic Selector Payload contains multiple proposals.
Test IKEv2.EN.I.1.1.1.3: Use of CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 1.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #4</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

**Part A (BASIC)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.

**Observable Results:**

*Part A*

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Possible Problems:**

- None.
Group 1.2. Use of Retransmission Timers

Test IKEv2.EN.I.1.1.2.1: Retransmissions of IKE_SA_INIT requests

Purpose:

To verify an IKEv2 device retransmits IKE_SA_INIT request using properly Header and Payloads format

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4
- [RFC 4718] - Sections 2.2 and 2.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set retransmission timer to 1 second.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 waits for the event of a timeout on NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 4: Judgment #2
The NUT retransmits an IKE_SA_INIT request which has the same Message ID value as the previous IKE_SA_INIT request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different retransmission timers. If it is impossible to configure the retransmission timer, modifying tester is required.
Test IKEv2.EN.I.1.1.2.2: Stop of retransmission of IKE_SA_INIT requests

Purpose:

To verify an IKEv2 device stops retransmission when it receives the corresponding response.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4
- [RFC 4718] - Sections 2.2 and 2.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set retransmission timer to 1 second.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
</table>

**Part A: (BASIC)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 waits for the event of a timeout on NUT.
4. Observe the messages transmitted on Link A.
5. TN1 responds with an IKE_SA_INIT response to the NUT.
6. TN1 waits for the event of a timeout on NUT.
7. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT retransmits an IKE_SA_INIT request which has the same Message ID value as the previous IKE_SA_INIT request’s Message ID value in IKE Header.

Step 7: Judgment #3
The NUT never retransmits an IKE_SA_INIT request which has the same Message ID value as the previous IKE_SA_INIT request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different retransmission timers. If it is impossible to configure the retransmission timer, modifying tester is required.
**Test IKEv2.EN.I.1.1.2.3: Retransmissions of IKE_AUTH requests**

**Purpose:**

To verify an IKEv2 device retransmits IKE_AUTH request using properly Header and Payloads format

**References:**

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

**Test Setup:**

- **Network Topology**
  Connect the devices according to the Common Topology.
- **Configuration**
  In each part, configure the devices according to the Common Configuration. In addition, set retransmission timer to 1 second.
- **Pre-Sequence and Cleanup Sequence**
  IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
</table>

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 waits for the event of a timeout on NUT.
6. Observe the messages transmitted on Link A.

**Observable Results:**
**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 6: Judgment #3**
The NUT retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

**Possible Problems:**

- Each NUT has the different retransmission timers. If it is imposibble to configure the retransmission timer, modifying tester is required.
Test IKEv2.EN.I.1.1.2.4: Stop of retransmission of IKE_AUTH requests

Purpose:
To verify an IKEv2 device stops retransmission when it receives the corresponding response.

References:
- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set retransmission timer to 1 second.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.

Packet #1  See Common Packet #2
Packet #2  See Common Packet #4
4. Observe the messages transmitted on Link A.
5. TN1 waits for the event of a timeout on NUT.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_AUTH response to the NUT.
8. TN1 waits for the event of a timeout on NUT.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

Step 9: Judgment #4
The NUT never retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different retransmission timers. If it is impossible to configure the retransmission timer, modifying tester is required.
Group 1.3. State Synchronization and Connection Timeouts

Test IKEv2.EN.I.1.1.3.1: State Synchronization with ICMP messages

Purpose:

To verify an IKEv2 device synchronizes its state when it receives ICMP messages.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #4</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>
Packet #4: ICMPv6 Destination Unreachable

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TR1's Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td>NUT's Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Code</td>
<td>0</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. After reception of an Echo Reply from NUT, TR1 transmits ICMP Destination Unreachable Message to the NUT and then TN1 transmits an Echo Request to the NUT.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Test IKEv2.EN.I.1.1.3.2: State Synchronization with IKE messages

Purpose:

To verify an IKEv2 device synchronizes its state when it receives IKE messages.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #4</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet # 20</td>
</tr>
</tbody>
</table>
Packet #3: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1's Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td>NUT's Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td>Destination Port</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE SA Initiator's SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE SA Responder's SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0–2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6–7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type</td>
<td>11 (INVALID_SPI)</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. TN1 transmits a cryptographically unprotected INFORMATIONAL request with Notify payload of type INVALID_SPI to the NUT.
9. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
10. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 10: Judgment #4**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.
Possible Problems:

- None
Test IKEv2.EN.I.1.1.3.3: Close connections when repeated attempts fail

Purpose:

To verify an IKEv2 device stops retransmission when it receives the corresponding response.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

| Step | Packet
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>2.</td>
<td>Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>3.</td>
<td>TN1 responds with an IKE_SA_INIT response to the NUT.</td>
</tr>
<tr>
<td>4.</td>
<td>Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>5.</td>
<td>TN1 waits for the event of a timeout on the NUT.</td>
</tr>
<tr>
<td>6.</td>
<td>Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>7.</td>
<td>Repeat Step 5 and Step 6 until the NUT’s last retransmission comes.</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
8. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE-SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

Step 8: Judgment #4
The NUT never retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

Possible Problems:

- None.
Test IKEv2.EN.I.1.1.3.4: Close connections when receiving INITIAL_CONTACT

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.1.3.5: Sending Liveness check

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.1.3.6: Sending Delete Payload for IKE_SA

Purpose:

To verify an IKEv2 device transmits a Delete Payload, when IKE_SA is deleted.

References:

- [RFC 4306] - Sections 2.4 and 3.11

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #4</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT
6. TN1 waits until expiring IKE_SA’s lifetime and does not respond to an INFORMATIONAL request with an INFORMATIONAL response for liveness check.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an INFORMATIONAL request with a Delete Payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.

**Possible Problems:**

- At Step 7, NUT can transmit INFORMATIONAL request with a Delete Payload including 2 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete CHILD_SA before transmitting an INFORMATIONAL request to delete IKE_SA.
Test IKEv2.EN.I.1.1.3.7: Sending Delete Payload for CHILD_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.1.3.8: Sending Liveness check with unprotected messages

This test case was deleted at revision 1.1.0.
Group 1.4. Version Numbers and Forward Compatibility

Test IKEv2.EN.I.1.1.4.1: Unrecognized payload types and Critical bit is not set

Purpose:
To verify an IKEv2 device ignores invalid payload types when the invalid type payload’s critical bit is not set.

References:
- [RFC 4306] - Sections 2.5

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #4: CREATE CHILD_Sa response

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>E payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Invalid Payload</td>
<td>Payload Length 4, Payload Type 41 (N)</td>
</tr>
<tr>
<td>N Payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>SA Payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>Ni, Nr payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
</tbody>
</table>

Part A: Invalid payload type 1 (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 1 and the invalid payload’s critical flag is not set.
11. Observe the messages transmitted on Link A.
12. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to NUT.
13. Observe the messages transmitted on Link A.

Part B: Invalid payload type 32 (BASIC)
14. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
15. Observe the messages transmitted on Link A.
16. TN1 responds with an IKE_SA_INIT response to the NUT.
17. Observe the messages transmitted on Link A.
18. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
19. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
20. Observe the messages transmitted on Link A.
21. Repeat Steps 19 and 20 until lifetime of SA is expired.
22. Observe the messages transmitted on Link A.
23. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 32 and the invalid payload’s critical flag is not set.
24. Observe the messages transmitted on Link A.
25. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to NUT.
26. Observe the messages transmitted on Link A.

Part C: Invalid payload type 49 (BASIC)
27. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
28. Observe the messages transmitted on Link A.
29. TN1 responds with an IKE_SA_INIT response to the NUT.
30. Observe the messages transmitted on Link A.
31. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
32. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
33. Observe the messages transmitted on Link A.
34. Repeat Steps 32 and 33 until lifetime of SA is expired.
35. Observe the messages transmitted on Link A.
36. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 49 and the invalid
payload’s critical flag is not set.
37. Observe the messages transmitted on Link A.
38. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to NUT.
39. Observe the messages transmitted on Link A.

Part D: Invalid payload type 255 (BASIC)
40. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
41. Observe the messages transmitted on Link A.
42. TN1 responds with an IKE_SA_INIT response to the NUT.
43. Observe the messages transmitted on Link A.
44. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
45. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
46. Observe the messages transmitted on Link A.
47. Repeat Steps 45 and 46 until lifetime of SA is expired.
48. Observe the messages transmitted on Link A.
49. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 255 and the invalid payload’s critical flag is not set.
50. Observe the messages transmitted on Link A.
51. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to NUT.
52. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 11: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 13: Judgment #5
The NUT transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

Part B

Step 15: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 17: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 20 Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 24: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 26: Judgment #5**
The NUT transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

**Part C**

**Step 28: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 30: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 33 Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 37: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 39: Judgment #5**
The NUT transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

**Part D**

**Step 41: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 43: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 46 Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 50: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 52: Judgment #5**
The NUT transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.1.4.2: Unrecognized payload types and Critical bit is set

Purpose:

To verify an IKEv2 device rejects the messages with invalid payload types when the invalid type payload’s critical bit is set.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1 See Common Packet #2
Packet #2 See Common Packet #4
Packet #3 See Common Packet #19
Packet #4 See below
Packet #5 See Common Packet #19

Packet #4: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>E payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Invalid Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Payload Length</td>
<td>41 (N)</td>
</tr>
<tr>
<td>Critical</td>
<td>1</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>N Payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>SA Payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>Ni, Nl payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>TSI Payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>All fields are same as Common Packet #14 Payload</td>
</tr>
</tbody>
</table>

Part A: Invalid payload type 1 and Critical bit is set (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 1 and the invalid payload’s critical flag is set.
11. Observe the messages transmitted on Link A.
12. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to NUT.
13. Observe the messages transmitted on Link A.

**Part B: Invalid payload type 32 and Critical bit is set (BASIC)**
14. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
15. Observe the messages transmitted on Link A.
16. TN1 responds with an IKE_SA_INIT response to the NUT.
17. Observe the messages transmitted on Link A.
18. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
19. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
20. Observe the messages transmitted on Link A.
21. Repeat Steps 19 and 20 until lifetime of SA is expired.
22. Observe the messages transmitted on Link A.
23. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 32 and the invalid payload’s critical flag is set.
24. Observe the messages transmitted on Link A.
25. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to NUT.
26. Observe the messages transmitted on Link A.

**Part C: Invalid payload type 49 and Critical bit is set (BASIC)**
27. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
28. Observe the messages transmitted on Link A.
29. TN1 responds with an IKE_SA_INIT response to the NUT.
30. Observe the messages transmitted on Link A.
31. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
32. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
33. Observe the messages transmitted on Link A.
34. Repeat Steps 32 and 33 until lifetime of SA is expired.
35. Observe the messages transmitted on Link A.
36. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 49 and the invalid...
payload’s critical flag is set.
37. Observe the messages transmitted on Link A.
38. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to NUT.
39. Observe the messages transmitted on Link A.

**Part D: Invalid payload type 255 and Critical bit is set (BASIC)**
40. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
41. Observe the messages transmitted on Link A.
42. TN1 responds with an IKE_SA_INIT response to the NUT.
43. Observe the messages transmitted on Link A.
44. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
45. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
46. Observe the messages transmitted on Link A.
47. Repeat Steps 45 and 46 until lifetime of SA is expired.
48. Observe the messages transmitted on Link A.
49. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 255 and the invalid payload’s critical flag is set.
50. Observe the messages transmitted on Link A.
51. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to NUT.
52. Observe the messages transmitted on Link A.

**Observation Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 11: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 13: Judgment #5**
The NUT never transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

**Part B**

**Step 15: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 17: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 20: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 24: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 26: Judgment #5**
The NUT never transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

**Part C**

**Step 28: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 30: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 33: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 37: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 39: Judgment #5**
The NUT never transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

**Part D**

**Step 41: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 43: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
**Step 46: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 50: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 52: Judgment #5**
The NUT never transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

**Possible Problems:**

- None.
Group 1.5. Cookies

Test IKEv2.EN.I.1.1.5.1: Retrying IKE_SA_INIT request with a Notify payload of type COOKIE

Purpose:

To verify an IKEv2 device retries IKE_SA_INIT request using a Notify payload of type COOKIE.

References:

- [RFC 4306] - Sections 2.6 and 3.10.1
- [RFC 4718] - Sections 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>All fields are same as Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IKEv2 Header</th>
<th>IKE_SA Initiator’s SPI</th>
<th>The same value as corresponding request’s IKE_SA Initiator’s SPI value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKE_SA Responder’s SPI</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
<td></td>
</tr>
<tr>
<td>Major Version</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Minor Version</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Exchange Type</td>
<td>34 (IKE_SA_INIT)</td>
<td></td>
</tr>
<tr>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>I (bit 3 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>V (bit 4 of Flags)</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
**Part A: (ADVANCED)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response including a Notify payload of type COOKIE to the NUT.
4. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_SA_INIT request including a Notify payload of type COOKIE containing following values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td>Message ID</td>
<td>0</td>
</tr>
<tr>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>N Payload Next Payload</td>
<td>0</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>COOKIE (16390)</td>
</tr>
<tr>
<td>Notification Data</td>
<td>Cookie value</td>
</tr>
</tbody>
</table>

**Figure 21 Notify Payload Format**

- A Next Payload field is set to SA Payload (33).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A SPI Size field is set to zero.
- A Notify Message Type field is set to COOKIE (16390).
- A Notification Data field is set to the TN1 supplied cookie data.
Possible Problems:

- None.
Test IKEv2.EN.I.1.1.5.2: Interaction of COOKIE and INVALID_KE_PAYLOAD

Purpose:

To verify an IKEv2 device properly handles a series of the Initial Exchanges using a Notify payload of type COOKIE and type INVALID_KE_PAYLOAD.

References:

- [RFC 4306] - Sections 2.6, 2.7 and 3.10.1
- [RFC 4718] - Sections 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, configure the IKE_SA parameters as described as following. KEi payload must carry either D-H Group 14 public key value or D-H Group 24 public key value.

<table>
<thead>
<tr>
<th>IKE SA Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption</td>
</tr>
<tr>
<td>Part A</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload: 41 (N)</td>
</tr>
<tr>
<td></td>
<td>Critical: 0 (No Next Payload)</td>
</tr>
</tbody>
</table>
Packet #2: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the common packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
</tbody>
</table>

N Payload

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>10</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>INVALID_KE_PAYLOAD (17)</td>
</tr>
<tr>
<td>Notification Data</td>
<td>The accepted D-H Group # (2)</td>
</tr>
</tbody>
</table>

Packet #3: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the common packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
</tbody>
</table>

N Payload

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>Any</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>COOKIE (16390)</td>
</tr>
<tr>
<td>Notification Data</td>
<td>Different cookie value from Packet #1's cookie value</td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response including a Notify payload of type COOKIE to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 responds with an IKE_SA_INIT response including a Notify payload of type INVALID KE_PAYLOAD to the NUT.
6. Observe the messages transmitted on Link A.
7. If the IKE_SA_INIT request from NUT includes a Notify payload of type COOKIE, TN1 responds with an IKE_SA_INIT response. The message has a different cookie value from the cookie value at Step 3.
   A) Observe the messages transmitted on Link A.
   B) TN1 responds with an IKE_SA_INIT response.
8. If the IKE_SA_INIT request from NUT does not include a Notify payload of type COOKIE, TN1 responds with an IKE_SA_INIT response.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload has D-H Group 14 public key value. Depending on configuration, it is possible to use D-H Group 24 for SA proposal and KEi payload instead of D-H Group 14.

**Step 4: Judgment #2**
The NUT transmits an IKE_SA_INIT request. The message has a Notify payload of type COOKIE with the cookie data supplied by the responder as the first payload. All other payloads are unchanged.

**Step 6: Judgment #3**
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. The message can have a Notify payload of type COOKIE with the cookie data supplied by the responder at Step 5. All other payloads are unchanged.

**Step 7A: Judgment #4**
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. The message must have a Notify payload of type COOKIE with the cookie data supplied by the responder at Step 7. All other payloads are unchanged.

**Step 9: Judgment #5**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.1.5.3: Interaction of COOKIE and INVALID_KE_PAYLOAD with unoptimized Responder

Purpose:

To verify an IKEv2 device properly handles a series of the Initial Exchanges using a Notify payload of type COOKIE and type INVALID_KE_PAYLOAD.

References:

- [RFC 4306] - Sections 2.6, 2.7 and 3.10.1
- [RFC 4718] - Sections 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, configure the IKE_SA parameters as described as following. KEi payload must carry either D-H Group 14 public key value or D-H Group 24 public key value.

<table>
<thead>
<tr>
<th>IKE_SA Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2, Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the common packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
</tbody>
</table>

N Payload:
- Next Payload: 0 (No Next Payload)
- Critical: 0
- Reserved: 0
- Payload Length: Any
- Protocol ID: 0
- SPI Size: 0
- Notify Message Type: COOKIE (16390)
- Notification Data: Cookie value

Packet #2: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the common packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
</tbody>
</table>

N Payload:
- Next Payload: 0 (No Next Payload)
- Critical: 0
- Reserved: 0

Packet #3: See below

Packet #4: See Common Packet #2
IPv6 FORUM TECHNICAL DOCUMENT

Part A: (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response including a Notify payload of type COOKIE to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 responds with an IKE_SA_INIT response including a Notify payload of type INVALID KE PAYLOAD to the NUT.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_SA_INIT response. The message has a different cookie value from the cookie value at Step 3.
8. Observe the messages transmitted on Link A.
9. TN1 responds with an IKE_SA_INIT response.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF HMAC_SHA1", "AUTH HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload has D-H Group 14 public key value. Depending on configuration, it is possible to use D-H Group 24 for SA proposal and KEi payload instead of D-H Group 14.

Step 4: Judgment #2
The NUT transmits an IKE_SA_INIT request. The message has a Notify payload of type COOKIE with the cookie data supplied by the responder as the first payload. All other payloads are unchanged.

Step 6: Judgment #3
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. The message can have a Notify payload of type COOKIE with the cookie data supplied by the responder at Step 5.

Step 8: Judgment #4
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. The message must have a Notify payload of type COOKIE with the cookie data supplied by the responder at Step 7. All other payloads are unchanged.

**Step 10: Judgment #5**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- None.
Group 1.6. Cryptographic Algorithm Negotiation

Test IKEv2.EN.I.1.1.6.1: Cryptographic Algorithm Negotiation for IKE_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-Shared key.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  From part A to part H, configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>IKE_SA_INIT exchanges Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_AES_CBC</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part B</td>
<td>DELETED</td>
<td>DELETED</td>
<td>DELETED</td>
<td></td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part D</td>
<td>ENCR_3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH_AES_XCBC_96</td>
<td></td>
</tr>
<tr>
<td>Part E</td>
<td>ENCR_3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 14</td>
</tr>
<tr>
<td>Part F</td>
<td>ENCR_3DES</td>
<td>PRF HMAC_SHA2_256</td>
<td>AUTH HMAC_SHA1_96</td>
<td></td>
</tr>
<tr>
<td>Part G</td>
<td>ENCR_3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA2_256_128</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part H</td>
<td>ENCR_3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA1_96</td>
<td>Group 24</td>
</tr>
</tbody>
</table>

*Italic* Pre-Sequence and Cleanup Sequence
IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1

See Common Packet #2
Part A: Encryption Algorithm ENCR_AES_CBC (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.

Part B: Encryption Algorithm ENCR_AES_CTR (ADVANCED)

This test case was deleted at revision 1.1.0.

Part C: PRF PRF_AES128_CBC (ADVANCED)
9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. TN1 responds with an IKE_SA_INIT response to the NUT.
12. Observe the messages transmitted on Link A.

Part D: Integrity Algorithm AUTH_AES_XCBC_96 (ADVANCED)
13. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. TN1 responds with an IKE_SA_INIT response to the NUT.
16. Observe the messages transmitted on Link A.

Part E: D-H Group Group 14 (ADVANCED)
17. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. TN1 responds with an IKE_SA_INIT response to the NUT.
20. Observe the messages transmitted on Link A.

Part F: PRF HMAC_SHA2_256 (ADVANCED)
21. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
22. Observe the messages transmitted on Link A.
23. TN1 responds with an IKE_SA_INIT response to the NUT.
24. Observe the messages transmitted on Link A.

Part G: Integrity Algorithm AUTH_HMAC_SHA2_256_128 (ADVANCED)
25. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. TN1 responds with an IKE_SA_INIT response to the NUT.
28. Observe the messages transmitted on Link A.

Part H: D-H Group Group 24 (ADVANCED)
29. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
30. Observe the messages transmitted on Link A.
31. TN1 responds with an IKE_SA_INIT response to the NUT.
32. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 4: Judgment #2
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 1.

Part B
This test case was deleted at revision 1.1.0.

Part C
Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_AES128_CBC", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 9.

Part D
Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_AES_XCBC_96" and "D-H Group 2" as proposed algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 13.

Part E
Step 18: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 14" as proposed algorithms.

Step 20: Judgment #2
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 17.

Part F
Step 22: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA2_256", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 24: Judgment #2
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 21.

Part G
Step 26: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA2_256_128" and "D-H Group 2" as proposed algorithms.
**Step 28: Judgment #2**
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 25.

**Part H**

**Step 30: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 24" as proposed algorithms.

**Step 32: Judgment #2**
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 29.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.1.6.2: Cryptographic Algorithm Negotiation for CHILD_SA

**Purpose:**

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-Shared key.

**References:**

- [RFC 4306] - Sections 2.7 and 3.3

**Test Setup:**

- Network Topology
  
  Connect the devices according to the Common Topology.

- Configuration
  
  From part A to part G, configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>Part</th>
<th>Encryption</th>
<th>Integrity</th>
<th>Extended Sequence Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ENCR AES CBC</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>B</td>
<td>ENCR AES CTR</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>C</td>
<td>ENCR NULL</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>D</td>
<td>ENCR 3DES</td>
<td>AUTH_AES_XCBC 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>E</td>
<td>ENCR 3DES</td>
<td>NONE</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>F</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC SHA1 96</td>
<td><em>Extended Sequence Numbers</em></td>
</tr>
<tr>
<td>G</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC SHA2 256 128</td>
<td>No Extended Sequence Numbers</td>
</tr>
</tbody>
</table>

- *Italic* parameters

**Procedure:**

IKEv2 on the NUT is disabled after each part.
Part A: Encryption Algorithm ENCR_AES_CBC (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.

Part B: Encryption Algorithm ENCR_AES_CTR (ADVANCED)
8. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
9. Observe the messages transmitted on Link A.
10. TN1 responds with an IKE_SA_INIT response to the NUT.
11. Observe the messages transmitted on Link A.
12. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
13. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
14. Observe the messages transmitted on Link A.

Part C: Encryption Algorithm ENCR_NULL (ADVANCED)
15. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
16. Observe the messages transmitted on Link A.
17. TN1 responds with an IKE_SA_INIT response to the NUT.
18. Observe the messages transmitted on Link A.
19. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
20. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
21. Observe the messages transmitted on Link A.

Part D: Integrity Algorithm AUTH_AES_XCBC_96 (ADVANCED)
22. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
23. Observe the messages transmitted on Link A.
24. TN1 responds with an IKE_SA_INIT response to the NUT.
25. Observe the messages transmitted on Link A.
26. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
27. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
28. Observe the messages transmitted on Link A.

Part E: Integrity Algorithm NONE (ADVANCED)
29. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
30. Observe the messages transmitted on Link A.
31. TN1 responds with an IKE_SA_INIT response to the NUT.
32. Observe the messages transmitted on Link A.
33. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
34. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
35. Observe the messages transmitted on Link A.
Part F: Extended Sequence Numbers (ADVANCED)
36. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
37. Observe the messages transmitted on Link A.
38. TN1 responds with an IKE_SA_INIT response to the NUT.
39. Observe the messages transmitted on Link A.
40. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
41. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
42. Observe the messages transmitted on Link A.

Part G: Integrity Algorithm AUTH_HMAC_SHA2_256_128 (ADVANCED)
43. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
44. Observe the messages transmitted on Link A.
45. TN1 responds with an IKE_SA_INIT response to the NUT.
46. Observe the messages transmitted on Link A.
47. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
48. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
49. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_AES_CBC", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Part B
Step 9: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 11: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_AES_CTR", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 14: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Part C
Step 16: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 18: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_NULL", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 21: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Part D

Step 23: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 25: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 28: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Part E

Step 30: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 32: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "NONE" and "No Extended Sequence Numbers" as proposed algorithms. However, the transform indicating "NONE" can be omitted.

Step 35: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Part F

Step 37: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 39: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1" and "Extended Sequence Numbers" as proposed algorithms.

Step 42: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Part G

Step 44: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 46: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA2_256_128" and "No Extended Sequence Numbers" as proposed algorithms.

Step 49: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Test IKEv2.EN.I.1.1.6.3: Sending Multiple Transforms for IKE_SA

Purpose:

To verify an IKEv2 device properly transmits IKE_SA_INIT request with multiple transforms for IKE_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration:

<table>
<thead>
<tr>
<th>IKE_SA_INIT exchanges Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td>ENCR_AES_CBC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRF_AES128_CBC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AUTH_AES_XCBC_96</td>
<td></td>
</tr>
<tr>
<td>Part D</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2, Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: Multiple Encryption Algorithms (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above.
2. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo-Random Functions (ADVANCED)

3. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above.
4. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithms (ADVANCED)

5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above.
6. Observe the messages transmitted on Link A.

**Part D: Multiple D-H Groups (ADVANCED)**

7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.

8. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Part B**

**Step 4: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "PRF_AES128_CBC""AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Part C**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "AUTH_AES_XCBC_96" and "D-H Group 2" as accepted algorithms.

**Part D**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as accepted algorithms. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.1.6.4: Sending Multiple Proposals for IKE_SA

Purpose:

To verify an IKEv2 device properly transmits IKE_AUTH request with multiple proposals for CHILD_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration.

<table>
<thead>
<tr>
<th>Proposal ID</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal #1</td>
<td>IKE ENCR_3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Proposal #2</td>
<td>IKE ENCR_AES_CBC</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_AES_XCBC_96</td>
<td>Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request with 2 SA Proposals.
SA Proposal #1 (ESP) includes "ENCR_3DES", "PRF HMAC_SHA1",
"AUTH HMAC_SHA1_96" and "D-H Group 2".
SA Proposal #2 (ESP) includes "ENCR_AES_CBC", "PRF_AES128_CBC", "AUTH_AES_XCBC_96" and "D-H Group 14". Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

Possible Problems:

- None.
Test IKEv2.EN.I.1.1.6.5: Sending Multiple Transforms for CHILD_SA

Purpose:

To verify an IKEv2 device properly transmits IKE_AUTH request with multiple transforms for CHILD_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration.

<table>
<thead>
<tr>
<th>IKE_AUTH exchanges Algorithms</th>
<th>Encryption</th>
<th>Integrity</th>
<th>ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td></td>
<td>ENCR_AES_CBC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTH_AES_XCBC_96</td>
<td></td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ESN</td>
<td></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: Multiple Encryption Algorithms (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above to the TN1.
2. Observe the messages transmitted on Link A.
3. NUT transmits an IKE_AUTH request including a SA payload as described above to the TN1.
4. Observe the messages transmitted on Link A.

**Part B: Multiple Integrity Algorithms (ADVANCED)**

5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above to the TN1.
6. Observe the messages transmitted on Link A.
7. NUT transmits an IKE_AUTH request including a SA payload as described above to the TN1.
8. Observe the messages transmitted on Link A.

**Part C: Multiple Extended Sequence Numbers (ADVANCED)**

9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above to the TN1.
10. Observe the messages transmitted on Link A.
11. NUT transmits an IKE_AUTH request including a SA payload as described above to the TN1.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "ENCR_AES_CBC", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers" and "Extended Sequence Number" as proposed algorithms.

**Possible Problems:**
• None.
Test IKEv2.EN.I.1.1.6.6: Sending Multiple Proposals for CHILD_SA

Purpose:

To verify an IKEv2 device properly transmits IKE_AUTH request with multiple proposals for CHILD_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration.

<table>
<thead>
<tr>
<th>IKE_AUTH exchanges Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal</td>
</tr>
<tr>
<td>Part A</td>
</tr>
<tr>
<td>Proposal #1</td>
</tr>
<tr>
<td>Proposal #2</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
</table>

Part A: (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" in SA Proposal #1 (ESP) and then "ENCR_AES_CBC", "AUTH_AES_XCBC_96" and "Extended Sequence Numbers" in SA Proposal #2 (ESP) as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.EN.I.1.1.6.7: Receipt of INVALID_KE_PAYLOAD

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA response with a Notify payload of type INVALID_KE_PAYLOAD.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration with enabling PFS by proposing D-H Group 2 and D-H Group 14 when rekeying. KEi payload must carry D-H Group 14 public key value in CREATE_CHILD_SA request. It is possible to use D-H Group 24 instead of D-H Group 14.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: See Common Packet #2
Packet #2: See Common Packet #4
Packet #3: See Common Packet #19
Packet #4: See below

Packet #4: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #14</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #14</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #14</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #14</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload: 0 (No Next Payload)</td>
</tr>
<tr>
<td></td>
<td>Critical: 0</td>
</tr>
<tr>
<td></td>
<td>Reserved: 0</td>
</tr>
<tr>
<td></td>
<td>Payload Length: 10</td>
</tr>
<tr>
<td></td>
<td>Protocol ID: 0</td>
</tr>
<tr>
<td></td>
<td>SPI Size: 0</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type: INVALID_KE_PAYLOAD (17)</td>
</tr>
<tr>
<td></td>
<td>Notification Data: The accepted D-H Group # (2)</td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response with a Notify payload of type INVALID_KE_PAYLOAD containing 2 (1024 Bit MODP) as Notification Data to the NUT.
11. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload must carry "D-H Group 14" public key value. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers", "D-H Group 2" and "D-H Group 14" as proposed algorithms and a Key Exchange payload which contains "D-H Group 2" public key value.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.1.6.8: Receipt of NO_PROPOSAL_CHOSEN

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.1.6.9: Response with inconsistent SA proposal for IKE_SA

Purpose:

To verify an IKEv2 device properly handles a response with a SA payload which is inconsistent with one of its proposals.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See below</th>
</tr>
</thead>
</table>

Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>SA Payload</td>
<td>See below</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>Ni Payload</td>
<td>Same as the Common Packet #2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>34 (KE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>Proposal #</td>
<td>SA Proposal</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Proposal Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1 (IKE)</td>
<td></td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT. But the response includes a SA payload which has a different Transform ID from the proposed one.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT never transmits an IKE_AUTH request.

Possible Problems:

- Step 4
  The NUT may transmit or retransmit an IKE_SA_INIT request.
Test IKEv2.EN.I.1.1.6.10: Response with inconsistent proposal for CHILD_SA

Purpose:

To verify an IKEv2 device properly handles a response with a SA payload which is inconsistent with one of its proposals.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH response

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
</tbody>
</table>
### IPv6 FORUM TECHNICAL DOCUMENT

<table>
<thead>
<tr>
<th>Part A: (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>3. TN1 responds with an IKE_SA_INIT response to the NUT.</td>
</tr>
<tr>
<td>4. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>5. TN1 responds with an IKE_AUTH response to the NUT. But the response includes a SA payload which has a different Transform ID from the proposed one.</td>
</tr>
<tr>
<td>6. TN1 transmits an Echo Request with IPsec ESP using ENCR_AES_CBC and AUTH_HMAC_SHA1_96.</td>
</tr>
<tr>
<td>7. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

### Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

---

<table>
<thead>
<tr>
<th>Payload</th>
<th>Next Payload</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
</tr>
<tr>
<td>SA Payload</td>
<td>See below</td>
<td></td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
</tr>
</tbody>
</table>

### Table of Payload Details

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Proposal Length</td>
<td>40</td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>3</td>
</tr>
</tbody>
</table>

| SA Transform | Next Payload | 0 (last) |
|--------------|--------------|
| Reserved | 0 |
| Transform Length | 8 |
| Transform Type | 3 (INTEG) |
| Reserved | 0 |
| Transform ID | 2 (HMAC_SHA1_96) |

| SA Attribute | Next Payload | 0 (last) |
|--------------|--------------|
| Reserved | 0 |
| Transform Length | 8 |
| Transform Type | 5 (Extended Sequence Number) |
| Reserved | 0 |
| Transform ID | 0 (No Extended Sequence Number) |

| Attribute Type | 14 (Key Length) |
| Attribute Value | 128 |

---

**IPv6 FORUM TECHNICAL DOCUMENT** 156  IPv6 Ready Logo Program IKEv2
Step 7: Judgment #3  
The NUT never transmits an Echo Reply with IPsec ESP using ENCR_AES_CBC and AUTH_HMAC_SHA1_96.

Possible Problems:

- Step 7  
The NUT may transmit or retransmit an IKE_AUTH request. And the NUT may notify INVALID_SPI.
Test IKEv2.EN.I.1.1.6.11: Receipt of INVALID_KE_PAYLOAD in Initial Exchange

Purpose:

To verify an IKEv2 device properly handles IKE_SA_INIT response with a Notify payload of type INVALID_KE_PAYLOAD.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.

- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, configure the IKE_SA parameters as described as following. KEi payload must carry D-H Group 14 public key value. It is possible to use D-H Group 24 instead of D-H Group 14.

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IKE_SA Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2, Group 14 or Group 24</td>
</tr>
</tbody>
</table>

Packet #1

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #2</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #2</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td>IKEv2 SA Responder’s SPI</td>
<td>See each Part</td>
</tr>
</tbody>
</table>

See below
<table>
<thead>
<tr>
<th>Critical</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>10</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>INVALID_KE_PAYLOAD (17)</td>
</tr>
<tr>
<td>Notification Data</td>
<td>The accepted D-H Group # (2)</td>
</tr>
</tbody>
</table>

Part A: IKE_SA Responder’s SPI is zero (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response including a Notify payload of type INVALID_KE_PAYLOAD containing 2 (1024 Bit MODP) as Notification Data to the NUT. The message’s IKE_SA Responder’s SPI is set to zero.
4. Observe the messages transmitted on Link A.

Part B: IKE_SA Responder’s SPI is not zero (ADVANCED)
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_SA_INIT response including a Notify payload of type INVALID_KE_PAYLOAD containing 2 (1024 Bit MODP) as Notification Data to the NUT. The message’s IKE_SA Responder’s SPI is set to one.
8. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload must carry "D-H Group 14" public key value. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

Step 4: Judgment #2
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. All other payloads are unchanged.

Part B

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload must carry "D-H Group 14" public key value. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

Step 4: Judgment #2
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. All other payloads are unchanged.

Possible Problems:

- None.
Test IKEv2.EN.I.1.1.6.12: Creating an IKE_SA without a CHILD_SA

Purpose:

To verify an IKEv2 device can handle a failure of creating a CHILD_SA during the IKE_AUTH exchange.

References:

- [RFC 4718] - Sections 4.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

Packet #4: IKE_AUTH response

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as Common Packet #4</td>
<td></td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #4</td>
<td></td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #4</td>
<td></td>
</tr>
<tr>
<td>Ext. Payload</td>
<td>Same as Common Packet #4</td>
<td></td>
</tr>
<tr>
<td>IPv6 Payload</td>
<td>Next Payload</td>
<td>39 (AUTH)</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
</tbody>
</table>
**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response with a Notify payload of type NO_PROPOSAL_CHOSEN to the NUT.
6. TN1 transmits an INFORMATIONAL request with no payloads to the NUT.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an INFORMATIONAL Response followed by an Encrypted payload with no payloads contained in it.

**Possible Problems:**

- None
Group 1.7. Traffic Selector Negotiation

Test IKEv2.EN.I.1.1.7.1: Narrowing the range of members of the set of traffic selectors

Purpose:

To verify an IKEv2 device allows the responder to choose a subset of the traffic proposed by the initiator.

References:

- [RFC4306] - Section 2.9

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
<th>Packet #2</th>
<th>See below</th>
</tr>
</thead>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 162 IPv6 Ready Logo Program IKEv2
Packet #2: IKE_AUTH response

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IP Protocol ID</td>
<td>6 (tcp)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IP Protocol ID</td>
<td>6 (tcp)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
</tbody>
</table>

Packet #3: TCP-SYN

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ESP</th>
<th>Security Parameter Index</th>
<th>CHILD_SA’s SPI value used by this message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
<td></td>
</tr>
<tr>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
<td></td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
<td></td>
</tr>
<tr>
<td>Next Header</td>
<td>6 (TCP)</td>
<td></td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td>The cryptographic checksum of the entire message</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TCP Header</th>
<th>Source Port</th>
<th>500</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Flags</td>
<td>SYN (0x02)</td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits a TCP-SYN packet with IPsec ESP using corresponding algorithms to closed port on NUT.
7. Observe the messages transmitted on Link A.
8. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
9. Observe the messages transmitted on Link A.

Observable Results:
Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCRYPTION_3DES", "PRF_HMAC_SHA1", "AUTHENTICATION_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCRYPTION_3DES", "AUTHENTICATION_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits a TCP-RST packet with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT never transmit an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Group 1.8. Error Handling

Test IKEv2.EN.I.1.1.8.1: INVALID_IKE_SPI

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.1.8.2: INVALID_SELECTORS

This test case was deleted at revision 1.1.0.
Group 1.10 Authentication of the IKE_SA

Test IKEv2.EN.I.1.1.10.1: Sending CERT Payload

Purpose:

To verify an IKEv2 device handles CERTREQ payload and transmits CERT payload properly.

References:

- [RFC 4306] - Sections 1.2 and 3.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Part</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>NUT’s global address on Link A</td>
</tr>
<tr>
<td>B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>nut.example.com</td>
</tr>
<tr>
<td>C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td><a href="mailto:nut@example.com">nut@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1

Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #2</td>
</tr>
</tbody>
</table>
KE Payload | Same as the Common Packet #2
---|---
Nr Payload | Next Payload 38 (CERTREQ)
| Other fields are same as the Common Packet #2
CERTREQ Payload | See below

<table>
<thead>
<tr>
<th>CERTREQ Payload</th>
<th>Next Payload</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>Any</td>
<td></td>
</tr>
<tr>
<td>Certificate Encoding</td>
<td>4 (X.509 Certificate – Signature)</td>
<td></td>
</tr>
<tr>
<td>Certificate Authority</td>
<td>any</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: ID_IPV6_ADDR (ADVANCED)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT request from the NUT, TN1 responds with an IKE_SA_INIT response with a CERTREQ payload to the NUT.
4. Observe the messages transmitted on Link A.

**Part B: ID_FQDN (ADVANCED)**
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. After reception of IKE_SA_INIT request from the NUT, TN1 responds with an IKE_SA_INIT response with a CERTREQ payload to the NUT.
8. Observe the messages transmitted on Link A.

**Part C: ID_RFC822_ADDR (ADVANCED)**
9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT request from the NUT, TN1 responds with an IKE_SA_INIT response with a CERTREQ payload to the NUT.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request. The request includes an ID payload with ID_IPV6_ADDR and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT's certificate as Certificate Data.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH request. The request includes an ID payload with ID_FQDN and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

Part C

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH request. The request includes an ID payload with ID_RFC822_ADDR and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

Possible Problems:

- None.
Test IKEv2.EN.I.1.1.10.2: Sending CERTREQ Payload

Purpose:

To verify an IKEv2 device transmits CERTREQ payload and handles CERT payload properly.

References:

- [RFC 4306] - Sections 1.2 and 3.7

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Remote</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>TN1’s global address on Link A</td>
</tr>
<tr>
<td>Part B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>tn.example.com</td>
</tr>
<tr>
<td>Part C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:tn@example.com">tn@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

**Part A: ID_IPV6_ADDR (ADVANCED)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.

**Part B: ID_FQDN (ADVANCED)**
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_SA_INIT response to the NUT.
8. Observe the messages transmitted on Link A.

**Part C: ID_RFC822_ADDR (ADVANCED)**
9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. TN1 responds with an IKE_SA_INIT response to the NUT.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.1.10.3: RSA Digital Signature

Purpose:
To verify an IKEv2 device authenticates the corresponding node by RSA Digital Signature.

References:
- [RFC 4306] - Sections 1.2 and 3.7

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Remote</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>TN1’s global address on Link A</td>
</tr>
<tr>
<td>Part B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>tn.example.com</td>
</tr>
<tr>
<td>Part C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:tn@example.com">tn@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #2: IKE_AUTH response

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 172 IPv6 Ready Logo Program IKEv2
IPv6 Header | Same as Common Packet #4
UDP Header  | Same as Common Packet #4
IKEv2 Header | Same as Common Packet #4
E Payload   | Same as Common Packet #4
IDr Payload | Next Payload | 37 (CERT)
            | Other fields are same as the Common Packet #4
CERT Payload | See below
AUTH Payload| Same as Common Packet #4
N Payload   | Same as Common Packet #4
SA Payload  | Same as Common Packet #4
TSi Payload | Same as Common Packet #4
TSr Payload | Same as Common Packet #4

CERT Payload | Next Payload | 39 (AUTH)
Critical     | 0
Reserved     | 0
Payload Length| Any
Certificate Encoding | 4 (X.509 Certificate – Signature)
Certificate Data  | TN1’s X.509 Certificate

Part A: ID_IPV6_ADDR (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response including an IDr payload as described above and a CERT payload to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.

Part B: ID_FQDN (ADVANCED)
8. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
9. Observe the messages transmitted on Link A.
10. TN1 responds with an IKE_SA_INIT response to the NUT.
11. Observe the messages transmitted on Link A.
12. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response including an IDr payload as described above and a CERT payload to the NUT.
13. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
14. Observe the messages transmitted on Link A.

Part C: ID_RFC822_ADDR (ADVANCED)
15. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
16. Observe the messages transmitted on Link A.
17. TN1 responds with an IKE_SA_INIT response to the NUT.
18. Observe the messages transmitted on Link A.
19. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response including an IDr payload as described above and a CERT payload to the NUT.
20. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
21. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Part B**

**Step 9: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 11: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Step 14: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Part C**

**Step 16: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 18: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Step 21: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.1.10.4: HEX string PSK

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 2.15

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Authentication Key Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote</td>
</tr>
<tr>
<td>0xabadcafeabadcafeabadcafeabadcafe (128 bit binary string)</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
</table>

Part A (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE\_SA\_INIT request including "ENCR\_3DES", "PRF\_HMAC\_SHA1", "AUTH\_HMAC\_SHA1\_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE\_AUTH request including "ENCR\_3DES", "AUTH\_HMAC\_SHA1\_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- None.
Group 1.11. Invalid values

Test IKEv2.EN.I.1.11.1: Non zero RESERVED fields in IKE_SA_INIT response

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>All RESERVED fields are set to one.</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response whose RESERVED fields are set to one to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.11.2: Non zero RESERVED fields in IKE_AUTH response

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #4</td>
</tr>
<tr>
<td>All RESERVED fields are set to one.</td>
<td></td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

Part A (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response whose RESERVED fields are set to one to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Possible Problems:

- None.
Test IKEv2.EN.I.1.1.11.3: Version bit is set

Purpose:
To verify an IKEv2 device ignores the content of Version bit in IKE messages.

References:
- [RFC 4306] - Sections 3.1

Test Setup:
- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version bit is set one.</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response whose Version bit is set to one to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

- None.
Test IKEv2.EN.I.1.11.4: Unrecognized Notify Message Type of Error

Purpose:

To verify an IKEv2 device ignores the unrecognized Notify Message Type intended for reporting error.

References:

- [RFC 4306] - Sections 3.10.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH response

<table>
<thead>
<tr>
<th></th>
<th>All fields are same as Common Packet #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>E Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
</tbody>
</table>
**IPv6 FORUM TECHNICAL DOCUMENT**

<table>
<thead>
<tr>
<th>Payload Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTH Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>N Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>SA Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Next Payload: 41 (Notify) Other fields are same as Common Packet #4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>N Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td>Procol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>16383</td>
</tr>
</tbody>
</table>

**Part A (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response with a Notify payload of unrecognized Notify Message Type value.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT never transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Possible Problems:**

- None.
Test IKEv2.EN.I.1.1.11.5: Unrecognized Notify Message Type of Status

Purpose:
To verify an IKEv2 device ignores the unrecognized Notify Message Type intended for reporting status.

References:
- [RFC 4306] - Sections 3.10.1

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>E Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Payload Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTH Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>N Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>SA Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>All fields are same as Common Packet #4</td>
</tr>
<tr>
<td>TSr payload</td>
<td>Next Payload 41 (Notify)</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload 0</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>65535</td>
</tr>
</tbody>
</table>

**Part A (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response with a Notify payload of unrecognized Notify Message Type value.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Possible Problems:**

- None.
Group 2. The CREATE_CHILD_SA Exchange

Group 2.1. Header and Payload Formats

Test IKEv2.EN.I.1.2.1.1: Sending CREATE_CHILD_SA request

Purpose:

To verify an IKEv2 device transmits CREATE_CHILD_SA request using properly Header and Payloads format.

References:

- [RFC 4306] - Sections 1.1.2, 1.2 and 3.3.2
- [RFC 4307] - Sections 3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: IKE Header Format (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired for 30 seconds.
9. Observe the messages transmitted on Link A.

Part B: Encrypted Payload Format (BASIC)
10. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an IKE_SA_INIT response to the NUT.
13. Observe the messages transmitted on Link A.
14. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
15. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
16. Observe the messages transmitted on Link A.
17. Repeat Steps 15 and 16 until lifetime of SA is expired for 30 seconds.
18. Observe the messages transmitted on Link A.
Part C: Notify Payload (REKEY_SA) Format (BASIC)
19. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. TN1 responds with an IKE_SA_INIT response to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
24. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
25. Observe the messages transmitted on Link A.
26. Repeat Steps 24 and 25 until lifetime of SA is expired for 30 seconds.
27. Observe the messages transmitted on Link A.

Part D: Notify Payload (USE_TRANSPORT_MODE) Format (BASIC)
28. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
29. Observe the messages transmitted on Link A.
30. TN1 responds with an IKE_SA_INIT response to the NUT.
31. Observe the messages transmitted on Link A.
32. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
33. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
34. Observe the messages transmitted on Link A.
35. Repeat Steps 33 and 34 until lifetime of SA is expired for 30 seconds.
36. Observe the messages transmitted on Link A.

Part E: SA Payload Format (BASIC)
37. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
38. Observe the messages transmitted on Link A.
39. TN1 responds with an IKE_SA_INIT response to the NUT.
40. Observe the messages transmitted on Link A.
41. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
42. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
43. Observe the messages transmitted on Link A.
44. Repeat Steps 42 and 43 until lifetime of SA is expired for 30 seconds.
45. Observe the messages transmitted on Link A.

Part F: Nonce Payload Format (BASIC)
46. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
47. Observe the messages transmitted on Link A.
48. TN1 responds with an IKE_SA_INIT response to the NUT.
49. Observe the messages transmitted on Link A.
50. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
51. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
52. Observe the messages transmitted on Link A.
53. Repeat Steps 51 and 52 until lifetime of SA is expired for 30 seconds.
54. Observe the messages transmitted on Link A.

Part G: TSi Payload Format (BASIC)
55. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
56. Observe the messages transmitted on Link A.
57. TN1 responds with an IKE_SA_INIT response to the NUT.
58. Observe the messages transmitted on Link A.
59. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
60. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
61. Observe the messages transmitted on Link A.
62. Repeat Steps 60 and 61 until lifetime of SA is expired for 30 seconds.
63. Observe the messages transmitted on Link A.

Part H: TSr Payload Format (BASIC)
64. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
65. Observe the messages transmitted on Link A.
66. TN1 responds with an IKE_SA_INIT response to the NUT.
67. Observe the messages transmitted on Link A.
68. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
69. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
70. Observe the messages transmitted on Link A.
71. Repeat Steps 69 and 70 until lifetime of SA is expired for 30 seconds.
72. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including properly formatted IKE Header containing following values:

![Figure 22 Header format](image-url)
• An IKE_SA Initiator’s SPI field is set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
• An IKE_SA Responder’s SPI field is set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
• A Next Payload field is set to Encrypted Payload (46).
• A Major Version field is set to 2.
• A Minor Version field is set to zero.
• An Exchange Type field is set to CREATE_CHILD_SA (36).
• A Flags field is set to (00010000)₂ = (16)₁₀.
• A Message ID field is set to the value incremented the previous IKE message’s Message ID by one.
• A Length field is set to the length of the message (header + payloads) in octets.

**Part B**

**Step 11: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 13: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 16: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 18: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including properly formatted Encrypted Payload containing following values:

![Figure 23 Encrypted payload]

- A Next Payload field is set to N Payload (41).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Check sum Data.
- An Initialization Vector field is set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
- An Encrypted IKE Payloads field is set to subsequent payloads encrypted by ENCR_3DES.
- A Padding field is set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
- A Pad Length field is set to the length of the Padding field.
- An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described inRFC.

**Part C**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 22: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 25: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 27: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including properly formatted Notify Payload containing following values:

![Figure 24 Notify Payload format](image)

- A Next Payload field is set to N Payload (41).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 12 bytes for this REKEY_SA.
- A Protocol ID field is set to ESP (3).
- A SPI Size field is set to the size of CHILD_SA Inbound SPI value to be rekeyed. It is 4 bytes for ESP.
Part D

Step 29: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 31: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 34: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 36: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including properly formatted Notify Payload containing following values:

- A Next Payload field is set to SA Payload (33).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 8 bytes for USE_TRANSPORT_MODE.
- A Protocol ID field is set to undefined (0).
- A SPI Size field is set to zero.
- A Notify Message Type field is set to USE_TRANSPORT_MODE (16391)

Part E

Step 38: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 40: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 43: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 45: Judgment #4**

![Figure 26 SA Payload contents](image)

The NUT transmits a CREATE_CHILD_SA request including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field is set to Ni Payload (40).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.

The following proposal must be included in Proposals field.
Figure 28 Proposal sub-structure format

Proposal #1
- A 0 or 2 field is set to zero if this structure is the last proposal, otherwise set to 2.
- A RESERED field is set to zero.
- A Proposal Length field is set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field is set to 1 if this structure is the first proposal, otherwise set to 1 greater than the previous proposal.
- A Protocol ID field is set to ESP (3).
- A SPI Size field is set to 4.
- A # of Transforms field is set to 3.
- A SPI field is set to the sending entity’s SPI (4 octets value)

Transform field is set to following (There are 3 Transform Structures).

Figure 29 Transform sub-structure format

Transform #1
- A 0 or 3 field is set to zero if this structure is the last proposal, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field is set to ENCR (1).
- A RESERVED field is set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field is set to zero if this structure is the last proposal, otherwise set to 3.
- A RESERVED field is set to zero.
• A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
• A Transform Type field is set to INTEG (3).
• A RESERVED field is set to zero.
• A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3
• A 0 or 3 field is set to zero if this structure is the last proposal, otherwise set to 3.
• A RESERVED field is set to zero.
• A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
• A Transform Type field is set to ESN (5).
• A RESERVED field is set to zero.
• A Transform ID set to No Extended Sequence Numbers (0).

Part F

Step 47: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 49: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 52: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 54: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including properly formatted Nonce Payload containing following values:

- A Next Payload field is set to TSi Payload (44).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Nonce Data field is set to random data generated by the transmitting entity.
- The size of the Nonce must between 16 and 256 octets.

Figure 30 Nonce Payload format

Part G

Step 56: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 58: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 61: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 63: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including properly formatted TSi Payload containing following values:

<table>
<thead>
<tr>
<th>Figure 31 TSi Payload format</th>
</tr>
</thead>
<tbody>
<tr>
<td>• A Next Payload field is set to TSr Payload (45).</td>
</tr>
<tr>
<td>• A Critical field is set to zero.</td>
</tr>
<tr>
<td>• A RESERVED field is set to zero.</td>
</tr>
<tr>
<td>• A Payload Length field is set to length of the current payload.</td>
</tr>
<tr>
<td>• A Number of TSs field is set to the number of actual traffic selectors.</td>
</tr>
<tr>
<td>• A RESERVED field is set to zero.</td>
</tr>
</tbody>
</table>

The following traffic selector must be included in Traffic Selectors field.

<table>
<thead>
<tr>
<th>Figure 32 Traffic Selector</th>
</tr>
</thead>
<tbody>
<tr>
<td>• A TS Type set to TS_IPV6_ADDR_RANGE (8).</td>
</tr>
</tbody>
</table>
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to less than or equal to NUT address.
- A Ending Address field is set to greater than or equal to NUT address.

**Part H**

**Step 65: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 67: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 70: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 72: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including properly formatted TSr Payload containing following values:

![Figure 33 TSr Payload format](image)

- A Next Payload field is set to zero.
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Number of TSs field is set to 1.
- A RESERVED field is set to zero.

The following traffic selector must be included in Traffic Selectors field.
Figure 34 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to less than or equal to TN1 address.
- An Ending Address field is set to less than or equal to TN1 address.

Possible Problems:

- The implementation may use different SA lifetimes by the implementation policy. In that case, the tester must change the expiration time to wait CREATE_CHILD_SA request.

- CREATE_CHILD_SA request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- The implementation may not set single proposal by the implementation policy. In this case, Security Association Payload contains multiple proposals.

- Each of transforms can be located in the any order.

- The implementation may not set single traffic selector by the implementation policy. In this case, Traffic Selector Payload contains multiple proposals.
Group 2.2. Use of Retransmission Timers

Test IKEv2.EN.I.1.2.2.1: Retransmissions of CREATE_CHILD_SA requests

Purpose:

To verify an IKEv2 device retransmits CREATE_CHILD_SA request using properly Header and Payloads format

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE-SA Lifetime to 300 seconds and set CHILD-SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE-SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE-SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. TN1 waits for the event of a timeout on NUT.
11. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 11: Judgment #5**
The NUT retransmits a CREATE_CHILD_SA request which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different lifetime of SA.
- Each NUT has the different retransmission timers.
Test IKEv2.EN.I.1.2.2.2: Stop of retransmission of CREATE_CHILD_SA requests

Purpose:

To verify an IKEv2 device stops retransmission when it receives the corresponding response.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. TN1 waits for the event of a timeout on NUT.
11. Observe the messages transmitted on Link A.
12. TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. TN1 waits for the event of a timeout on NUT.
14. Observe the messages transmitted on Link A.

Observable Results:

Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 11: Judgment #5**
The NUT retransmits a CREATE_CHILD_SA request which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

**Step 14: Judgment #6**
The NUT stops the retransmissions of a CREATE_CHILD_SA request which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different lifetime of SA.
- Each NUT has the different retransmission timers.
Group 2.3. Rekeying CHILD_SAs Using a CREATE_CHILD_SA exchange

Test IKEv2.EN.I.1.2.3.1: Close the replaced CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the CREATE_CHILD_SA Exchanges to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1 See Common Packet #2
Packet #2 See Common Packet #4
Packet #3 See Common Packet #19
Packet #4 See Common Packet #14

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an INFORMATIONAL response with a Delete payload to the NUT.
13. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to the NUT.
14. Observe the messages transmitted on Link A.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 11: Judgment #5
The NUT transmits an INFORMATIONAL request with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the inbund SPI value to be deleted as SPI.

Step 14: Judgment #6
The NUT transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.3.2: Use of the new CHILD_SA

Purpose:

To verify an IKEv2 device properly rekeys CHILD_SA

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

Packet #5: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
</tbody>
</table>

This packet is cryptographically protected by the new CHILD_SA negotiated at Step 10.
Exchange Type | 37 (INFORMATIONAL)
---|---
X (bits 0-2 of Flags) | 0
I (bit 3 of Flags) | any
V (bit 4 of Flags) | 0
R (bit 5 of Flags) | 1
X (bits 6-7 Flags) | 0
Message ID | 0
Length | any

E Payload
- Next Payload | 42 (D)
- Critical | 0
- Reserved | 0
- Payload Length | any
- Initialization Vector | The same value as block length of the underlying encryption algorithm
- Encrypted IKE Payloads | Subsequent payloads encrypted by underlying encryption algorithm
- Padding | Any value which to be a multiple of the encryption block size
- Pad Length | The length of the Padding field
- Integrity Checksum Data | The Cryptographic checksum of the entire message

D Payload
- Next Payload | 0
- Critical | 0
- Reserved | 0
- Payload Length | 12
- Protocol ID | 3 (ESP)
- # of SPIs | 4
- Security Parameter Index | NUT’s inbound CHILD_SA SPI value to be deleted

### Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an INFORMATIONAL response with a Delete payload to the NUT.
13. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to the NUT.
14. Observe the messages transmitted on Link A.

### Observable Results:

#### Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 11: Judgment #5
The NUT transmits an INFORMATIONAL request with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the inblund SPI value to be deleted as SPI.

Step 14: Judgment #6
The NUT transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.3.3: Lifetime of CHILD_SA expires

Purpose:

To verify an IKEv2 device properly recognizes the lifetime of CHILD_SAs.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>See Common Packet #2</td>
<td>See Common Packet #4</td>
</tr>
<tr>
<td>Packet #3</td>
<td>Packet #4</td>
</tr>
<tr>
<td>See Common Packet #19</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. TN1 waits for the event of a timeout on the NUT.
9. After timeout of CHILD_SA on the NUT, TN1 transmits an Echo Request with IPsec ESP which has expired to the NUT.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #4
The NUT does not transmit an Echo Reply with IPsec ESP using already expired CHILD_SA.

Possible Problems:
- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.3.4: Sending Multiple Transform

Purpose:

To verify an IKEv2 device properly transmits CREATE_CHILD_SA request with multiple transforms to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration:

  | CREATE_CHILD_SA exchanges | Algorithms
  |---------------------------|-------------
  | Encryption | Integrity | ESN |
  | Part A | ENCR_3DES | AUTH_HMAC_SHA1_96 | No ESN |
  | Part B | ENCR_3DES | AUTH_HMAC_SHA1_96 | No ESN |
  | Part C | ENCR_3DES | AUTH_HMAC_SHA1_96 | No ESN |

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: Multiple Encryption Algorithms (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired for 30 seconds.
9. Observe the messages transmitted on Link A.

Part B: Multiple Integrity Algorithms (ADVANCED)
10. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an IKE_SA_INIT response to the NUT.
13. Observe the messages transmitted on Link A.
14. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
15. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
16. Observe the messages transmitted on Link A.
17. Repeat Steps 15 and 16 until lifetime of SA is expired for 30 seconds.
18. Observe the messages transmitted on Link A.
Part C: Multiple Extended Sequence Numbers (ADVANCED)

19. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. TN1 responds with an IKE_SA_INIT response to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
24. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
25. Observe the messages transmitted on Link A.
26. Repeat Steps 24 and 25 until lifetime of SA is expired for 30 seconds.
27. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "ENCR_AES_CBC", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Part B

Step 11: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 13: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 16: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 18: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.
Part C

Step 20: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 22: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 25: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 27: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers" and "Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.3.5: Sending Multiple Proposal

Purpose:

To verify an IKEv2 device properly transmits CREATE_CHILD_SA request with multiple proposals to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration:

<table>
<thead>
<tr>
<th>Part A</th>
<th>Proposal 1</th>
<th>Protocol ID</th>
<th>Encryption</th>
<th>Integrity</th>
<th>ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Proposal #1</td>
<td>ESP</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC_SHA1 96</td>
<td>No ESN</td>
</tr>
<tr>
<td></td>
<td>Proposal #2</td>
<td>ESP</td>
<td>ENCR_AES_CBC</td>
<td>AUTH_AES_XCBC_96</td>
<td>ESN</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired for 30 seconds.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" in SA Proposal #1 (ESP) and then "ENCR_AES_CBC", "AUTH_AES_XCBC_96" and "Extended Sequence Numbers" in SA Proposal #2 (ESP) as accepted algorithms.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.3.6: Rekeying Failure

Purpose:

To verify an IKEv2 device properly handles rekeying failure.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 30 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #4</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #14</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying IKE_SA from the NUT, TN1 rejects the NUT’s proposal. TN1 responds with a CREATE_CHILD_SA response with a Notify of type NO_PROPOSAL_CHOSEN.
11. TN1 transmits an INFORMATIONAL request for liveness check to the NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request for rekeying IKE_SA.
The request includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 12: Judgment #5
The NUT never responds with an INFORMATIONAL response to an INFORMATIONAL request.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.3.7: Perfect Forward Secrecy

Purpose:
To verify an IKEv2 device properly rekeys CHILD_SA when Perfect Forward Secrecy enables.

References:
- [RFC 4306] - Sections 2.12

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds. Enable PFS.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: See Common Packet #2
Packet #2: See Common Packet #4
Packet #3: See Common Packet #19
Packet #4: See below
Packet #5: See below
Packet #6: See Common Packet #19

This packet is cryptographically protected by the new CHILD_SA negotiated at Step 10.

Packet #4: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>Payload Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #14</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #14</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #14</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #14</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #14</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #14</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #14</td>
</tr>
<tr>
<td>Nr Payload</td>
<td>Next Payload 34 (KE)</td>
</tr>
<tr>
<td>KeR Payload</td>
<td>Next Payload 44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

Packet #5: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #18</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #18</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #18</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #18</td>
</tr>
<tr>
<td>Delete Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>42 (Delete)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>12</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of SPIs</td>
<td>1</td>
</tr>
<tr>
<td>Security Parameter Index(es) (SPI)</td>
<td>SPI negotiated by Initial Exchange</td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an INFORMATIONAL response with a Delete payload to the NUT.
13. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to the NUT.
14. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.
Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENC3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 11: Judgment #5
The NUT transmits an INFORMATIONAL request with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the inbund SPI value to be deleted as SPI.

Step 14: Judgment #6
The NUT transmits an Echo Reply with IPsec ESP using the second negotiated algorithms.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.3.8: Use of the old CHILD_SA

Purpose:

To verify an IKEv2 device properly handles new CHILD_SA and old CHILD_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

Packet #1 See Common Packet #2
Packet #2 See Common Packet #4
Packet #3 See Common Packet #19
Packet #4 See Common Packet #14
Packet #5 See Common Packet #19

This packet is cryptographically protected by the new CHILD_SA negotiated at Step 5.

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
11. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms again.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 12: Judgment #5
The NUT transmits an Echo Reply with IPsec ESP. The NUT can use both the first CHILD_SA and the new CHILD_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Group 2.4. Rekeying IKE_SAs Using a CREATE_CHILD_SA exchange

Test IKEv2.EN.I.1.2.4.1: Close the replaced IKE_SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds.
with a CREATE_CHILD_SA response to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an INFORMATIONAL response to close the replaced IKE_SA.
13. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms inherited from the replaced IKE_SA.
14. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA's SPI value in the SPI field.

Step 11: Judgment #5
The NUT transmits an INFORMATIONAL request with a Delete payload to close the replaced IKE_SA.

Step 14: Judgment #6
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms inherited from the replaced IKE_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.4.2: Use of the new IKE_SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds.
with a CREATE_CHILD_SA response to the NUT.

11. Observe the messages transmitted on Link A.
12. TN1 responds with an INFORMATIONAL response to an INFORMATIONAL request to close the replaced IKE_SA.
13. TN1 transmits an INFORMATIONAL request with no payloads cryptographically protected by new IKE_SA.
14. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

**Step 11: Judgment #5**
The NUT transmits an INFORMATIONAL request with a Delete payload to close the replaced IKE_SA.

**Step 14: Judgment #6**
The NUT responds with an INFORMATIONAL response with not payloads cryptographically protected by new IKE_SA.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.4.3: Lifetime of IKE_SA expires

Purpose:

To verify an IKEv2 device properly recognizes the lifetime of IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #4</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 236 IPv6 Ready Logo Program IKEv2
Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an INFORMATIONAL request with no payloads to the NUT.
7. Observe the messages transmitted on Link A.
8. TN1 waits for the event of a timeout on the NUT.
9. After timeout of CHILD_SA on the NUT, TN1 transmits an INFORMATIONAL request with no payloads using already expired IKE_SA.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT responds with an INFORMATIONAL response with no payloads.

Step 10: Judgment #4
The NUT does not respond with an INFORMATIONAL response with no payloads using already expired IKE_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.4.4: Sending Multiple Transform

Purpose:

To verify an IKEv2 device properly transmits CREATE_CHILD_SA request with multiple transforms to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.

<table>
<thead>
<tr>
<th>CREATE_CHILD_SA exchanges Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part D</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2, Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: Multiple Encryption Algorithms (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo-Random Functions (ADVANCED)
10. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an IKE_SA_INIT response to the NUT.
13. Observe the messages transmitted on Link A.
14. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
15. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
16. Observe the messages transmitted on Link A.
17. Repeat Steps 6 and 7 until lifetime of SA is expired.
18. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithms (ADVANCED)
19. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. TN1 responds with an IKE_SA_INIT response to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
24. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
25. Observe the messages transmitted on Link A.
26. Repeat Steps 6 and 7 until lifetime of SA is expired.
27. Observe the messages transmitted on Link A.

Part D: Multiple D-H Groups (ADVANCED)
28. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
29. Observe the messages transmitted on Link A.
30. TN1 responds with an IKE_SA_INIT response to the NUT.
31. Observe the messages transmitted on Link A.
32. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
33. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
34. Observe the messages transmitted on Link A.
35. Repeat Steps 6 and 7 until lifetime of SA is expired.
36. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Part B

Step 11: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 13: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Step 16: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 18: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "PRF_AES128_CBC", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Part C
Step 20: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 22: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 25: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 27: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "AUTH_AES_XCBC_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Part D
Step 29: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 31: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 34: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 36: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.
And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.4.5: Sending Multiple Proposal

Purpose:

To verify an IKEv2 device properly transmits CREATE_CHILD_SA request with multiple proposal to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
  - In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.

<table>
<thead>
<tr>
<th>Proposal</th>
<th>Protocol ID</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>Proposal #1</td>
<td>IKE</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
</tr>
<tr>
<td></td>
<td>Proposal #2</td>
<td>IKE</td>
<td>ENCR_AES_CBC</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_AES_XCBC_96</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: Multiple Encryption Algorithms (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request with 2 SA Proposals.
SA Proposal #1 (ESP) includes "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2".
SA Proposal #2 (ESP) includes "ENCR_AES_CBC", "PRF_AES128_CBC", "AUTH_AES_XCBC_96" and "D-H Group 14". Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.4.6: Use of the old IKE_SA

Purpose:

To verify an IKEv2 device properly handles new CHILD_SA and old CHILD_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
11. TN1 transmits an INFORMATIONAL request with no payload to the NUT. The message is encrypted by the old IKE_SA.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 12: Judgment #5
The NUT transmits an INFORMATIONAL response with no payload to the TN1. The message is encrypted by the old IKE_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.4.7: Changing PRFs when rekeying the IKE_SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8
- [RFC 4718] - Sections 5.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
  Configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>IKE_SA Rekeying Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption</td>
</tr>
<tr>
<td>Part A</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #4: CREATE_CHILD_SA response
Packet #4 is same as Common Packet #12 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform TypePRF replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>8</td>
<td>2 (PRF)</td>
<td>0</td>
<td>4 (PRF_AES128_XCBC)</td>
</tr>
</tbody>
</table>
Part A: (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an INFORMATIONAL response to an INFORMATIONAL request to close the replaced IKE_SA.
13. TN1 transmits an INFORMATIONAL request with no payloads cryptographically protected by new IKE_SA.
14. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_AES128_XCBC", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Step 11: Judgment #5
The NUT transmits an INFORMATIONAL request with a Delete payload to close the replaced IKE_SA.

Step 14: Judgment #6
The NUT responds with an INFORMATIONAL response with not payloads cryptographically protected by new IKE_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Group 2.5. Creating New CHILD_SAs with the CREATE_CHILD_SA Exchanges

Test IKEv2.EN.I.1.2.5.1: Create new CHILD_SA by sending CREATE_CHILD_SA request

Purpose:
To verify an IKEv2 device properly handles the CREATE_CHILD_SA Exchanges to generate new CHILD_SAs.

References:
- [RFC 4306] - Sections 1.1.2, 1.2 and 3.3.2
- [RFC 4307] - Sections 3
- [RFC 4718] - Sections 4.1

Test Setup:
- Network Topology: Connect the devices according to the Common Topology.
- Configuration: In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence: IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See below</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #4</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH response
<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #4</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #4</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See below</td>
</tr>
</tbody>
</table>

Other fields are same as the Common Packet #4

**TS Payload**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
</tr>
<tr>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td></td>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

**TSr Payload**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
</tr>
<tr>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td></td>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. NUT starts to negotiate new CHILD_SA with TN1 by sending CREATE_CHILD_SA request.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Possible Problems:

- None.
Test IKEv2.EN.I.1.2.5.2: Receipt of cryptographically valid message on the new SA

Purpose:

To verify an IKEv2 device properly handles the CREATE_CHILD_SA Exchanges to generate new CHILD_SAs.

References:

- [RFC 4306] - Sections 1.1.2, 1.2 and 3.3.2
- [RFC 4307] - Sections 3
- [RFC 4718] - Sections 4.1

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #19</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #7</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

- Packet #2: IKE_AUTH response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Traffic Selector</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPv6 ADDR RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
</tbody>
</table>

- **Packet #3: TCP SYN packet**

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>ESP Security Parameter Index</td>
<td>CHILD_SA’s SPI value used by this message</td>
<td></td>
</tr>
<tr>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
<td></td>
</tr>
<tr>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
<td></td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
<td></td>
</tr>
<tr>
<td>Next Header</td>
<td>6 (TCP)</td>
<td></td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC</td>
<td></td>
</tr>
</tbody>
</table>

| TCP Header | Source Port | 30000 |
|           | Destination Port | 30000 |
|           | Flags | SYN (0x02) |

- **Packet #5: CREATE_CHILD_SA response**

| IPv6 Header | Same as the Common Packet #8 |
| UDP Header | Same as the Common Packet #8 |
| IKEv2 Header | Same as the Common Packet #8 |
| E Payload | Same as the Common Packet #8 |
| ID Payload | Same as the Common Packet #8 |
| AUTH Payload | Same as the Common Packet #8 |
| N Payload | Same as the Common Packet #8 |
| SA Payload | Same as the Common Packet #8 |
| TSi Payload | Other fields are same as the Common Packet #8 |
| TSr Payload | Other fields are same as the Common Packet #8 |

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPv6 ADDR RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IP Protocol ID</td>
<td>58 (IPv6-ICMP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Starting Address</td>
<td>NUT’s Global Address on Link X</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ending Address</td>
<td>NUT’s Global Address on Link X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPv6 ADDR RANGE)</th>
</tr>
</thead>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

258

IPv6 Ready Logo Program IKEv2

IPv6 FORUM

IPv6 Protocol ID 58 (IPV6-ICMP)
Selector Length 40
Start Port 0
End Port 65535
Starting Address TN1’s Global Address on Link A
Ending Address TN1’s Global Address on Link A

● Packet #6: TCP SYN packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ESP</th>
<th>Security Parameter Index</th>
<th>CHILD_SA’s SPI value used by this message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
<td></td>
</tr>
<tr>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
<td></td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
<td></td>
</tr>
<tr>
<td>Next Header</td>
<td>6 (TCP)</td>
<td></td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TCP Header</th>
<th>Source Port</th>
<th>30000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Port</td>
<td>30000</td>
<td></td>
</tr>
<tr>
<td>Flags</td>
<td>SYN (0x02)</td>
<td></td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits a TCP-SYN packet with IPsec ESP using corresponding algorithms to closed port 30000 on NUT.
7. Observe the messages transmitted on Link A.
8. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
9. Observe the messages transmitted on Link A.
10. NUT starts to negotiate new CHILD_SA with TN1 by sending CREATE_CHILD_SA request.
11. Observe the messages transmitted on Link A.
12. After a reception of CREATE_CHILD_SA request from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. TN1 transmits a TCP-SYN packet with IPsec ESP using corresponding algorithms to closed port 30000 on NUT.
14. Observe the messages transmitted on Link A.
15. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
16. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENC_R_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits a TCP-RST packet with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT never transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 14: Judgment #6**
The NUT transmits a TCP-RST packet with IPsec ESP using corresponding algorithms.

**Step 16: Judgment #7**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Possible Problems:**

- If the NUT uses TCP port 30000 for other applications, the TN1 transmits TCP-SYN packets to other closed TCP port on the NUT.
Group 2.6. Exchange Collisions

Test IKEv2.EN.I.1.2.6.1: Simultaneous CHILD_SA Close

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.2: Simultaneous IKE_SA Close

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.3: Simultaneous CHILD_SA Rekeying

Purpose:

To verify an IKEv2 device properly handles simultaneous CREATE_CHILD_SA Exchanges to rekey CHILD_SA.

References:

- [RFC 4718] - Sections 5.11.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
### IPv6 FORUM TECHNICAL DOCUMENT

**IPv6 Ready Logo Program IKEv2**

<table>
<thead>
<tr>
<th>Packet #</th>
<th>See Common Packet #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #1</td>
<td>Packet #2</td>
</tr>
<tr>
<td>Packet #2</td>
<td>Packet #4</td>
</tr>
<tr>
<td>Packet #3</td>
<td>Packet #19</td>
</tr>
<tr>
<td>Packet #4</td>
<td>Packet #13</td>
</tr>
<tr>
<td>Packet #5</td>
<td>Packet #14</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #7</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #8</td>
<td>Packet #19</td>
</tr>
</tbody>
</table>
### Packet #6: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TN1’s Global Address on Link X</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP Header</th>
<th>Source Port</th>
<th>Destination Port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500</td>
<td>500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IKEv2 Header</th>
<th>IKE_SA Initiator’s SPI</th>
<th>any</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0–2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6–7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request’s Message ID</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
</tbody>
</table>

| E Payload | Next Payload | 42 (D) |
|           | Critical     | 0     |
|           | Reserved     | 0     |
|           | Payload Length | any |
|           | Initialization Vector | The same value as block length of the underlying encryption algorithm |
|           | Encrypted IKE Payloads | Subsequent payloads encrypted by underlying encryption algorithm |
|           | Padding      | Any value which to be a multiple of the encryption block size |
|           | Pad Length   | The length of the Padding field |

| D Payload | Next Payload | 0 |
|           | Critical     | 0 |
|           | Reserved     | 0 |
|           | Payload Length | 12 |
|           | Protocol ID  | 3 (ESP) |
|           | SPI Size     | 4 |
|           | # of SPIs    | 1 |
|           | Security Parameter Index | NUT’s inbound CHILD_SA SPI value of the original CHILD_SA |

### Packet #7: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TN1’s Global Address on Link X</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP Header</th>
<th>Source Port</th>
<th>Destination Port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500</td>
<td>500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IKEv2 Header</th>
<th>IKE_SA Initiator’s SPI</th>
<th>any</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0–2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6–7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request’s Message ID</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
</tbody>
</table>

| E Payload | Next Payload | 42 (D) |
|           | Critical     | 0     |
|           | Reserved     | 0     |
|           | Payload Length | any |
|           | Initialization Vector | The same value as block length of the underlying encryption algorithm |
|           | Encrypted IKE Payloads | Subsequent payloads encrypted by underlying encryption algorithm |
|           | Padding      | Any value which to be a multiple of the encryption block size |
Pad Length | The length of the Padding field
---|---
Next Payload | 0
Critical | 0
Reserved | 0
Payload Length | 12
Protocol ID | 3 (ESP)
SPI Size | 4
# of SPIs | 1
Security Parameter Index | NUT’s inbound CHILD_SA SPI value of the new CHILD_SA initiated by the NUT at Step 9

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. TN1 transmits a CREATE_CHILD_SA request to rekey CHILD_SA to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 responds with a CREATE_CHILD_SA response to the CREATE_CHILD_SA received at Step 9. The response message includes minimum Nonce Data.
13. Observe the messages transmitted on Link A.
14. TN1 responds with an INFORMATIONAL response to the INFORMATIONAL request received at Step 13.
15. Observe the messages transmitted on Link A.
16. TN1 responds with an INFORMATIONAL response to the INFORMATIONAL request received at Step 15.
17. TN1 transmits an Echo Request with IPsec ESP using the existing algorithms to the NUT.
18. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request to rekey a CHILD_SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers".
Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 13: Judgment #6**
The NUT transmits an INFORMATIONAL request with a Delete Payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the inblund SPI value of the original CHILD_SA.

**Step 15: Judgment #7**
The NUT transmits an INFORMATIONAL request with a Delete Payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the inblund SPI value of the new CHILD_SA initiated by the NUT at Step 9.

**Step 18: Judgment #8**
The NUT transmits an Echo Reply with IPsec ESP using the existing CHILD_SA initiated by the TN1 at Step 10.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.6.4: Simultaneous CHILD_SA Rekeying with retransmission

Purpose:

To verify an IKEv2 device properly handles simultaneous CREATE_CHILD_SA Exchanges to rekey CHILD_SA.

References:

- [RFC 4718] - Sections 5.11.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #5: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1's Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP Header</th>
<th>Source Port</th>
<th>500</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Packet 1: See Common Packet #2
Packet 2: See Common Packet #4
Packet 3: See Common Packet #19
Packet 4: See Common Packet #13
Packet 5: See below
Packet 6: See below
Packet 7: See Common Packet #19
### Packet #6: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #14</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #14</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #14</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #14</td>
</tr>
</tbody>
</table>
| N Payload | Next Payload 0  
| Critical | 0  
| Reserved | 0  
| Payload Length | 8  
| Protocol ID | 0  
| SPI Size | 0  
| Notify Message Type | NO_PROPOSAL_CHOSEN (14) |

### Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. TN1 transmits a CREATE_CHILD_SA request to rekey CHILD_SA to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 transmits an INFORMAITONAL request with a Delete Payload to close the replaced CHILD_SA.
13. Observe the messages transmitted on Link A.
14. Observe the messages transmitted on Link A.
15. TN1 responds with a CREATE_CHILD_SA response with a Notify payload of type NO_PROPOSAL_CHOSEN to the retransmitted CREATE_CHILD_SA request.
16. TN1 transmits an Echo Request with IPsec ESP using the existing algorithms to the NUT.
17. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request to rekey a CHILD_SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 13: Judgment #6**
The NUT transmits an INFORMATIONAL response with a Delete Payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the inbund SPI value of the original CHILD_SA.

**Step 14: Judgment #7**
The NUT retransmits the same CREATE_CHILD_SA request as the message at Step 11. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 17: Judgment #8**
The NUT transmits an Echo Reply with IPsec ESP using the existing CHILD_SA initiated by the TN1 at Step 10.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.6.5: Simultaneous IKE_SA Rekeying

Purpose:
To verify an IKEv2 device properly handles a CREATE_CHILD_SA to rekey IKE_SA.

References:
- [RFC 4718] - Sections 5.11.4

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1 See Common Packet #2
Packet #2 See Common Packet #4
Packet #3 See Common Packet #19
Packet #4 See Common Packet #11
Packet #5 See Common Packet #12
Packet #6 See Common Packet #18
Packet #7 See Common Packet #18
Packet #8 See Common Packet #17
**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. TN1 transmits a CREATE_CHILD-SA request to rekey IKE_SA to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 responds with a CREATE_CHILD-SA response to the CREATE_CHILD-SA request received at Step 9. The response message includes minimum Nonce Data to make the NUT send a message to close duplicated IKE_SA.
13. Observe the messages transmitted on Link A.
14. TN1 responds with an INFORMATIONAL response with no payload.
15. Observe the messages transmitted on Link A.
16. TN1 responds with an INFORMATIONAL response with no payload.
17. TN1 transmits an INFORMATIONAL request with no payload to the NUT. The message is cryptographically protected by the new IKE_SA initiated by TN1 at Step 10.
18. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request to rekey an IKE_SA. The message includes "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request has a SA payload including 1 (IKE) in the Protocol ID field, 8 in the SPI size field and new IKE_SA’s SPI value in the SPI field.

**Step 11: Judgment #5**
The NUT responds a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the proposal in the SA payload Response has a SA payload including 1 (IKE) in the Protocol ID field, 8 in the SPI size field and new IKE_SA’s responder’s SPI value in the SPI field.
Step 13: Judgment #6
The NUT transmits an INFORMATIONAL request. The message’s IKE_SA Initiator’s SPI value is the IKE_SA Initiator’s SPI value of the original IKE_SA, and the message’s IKE_SA Responder’s SPI value is the IKE_SA Responder’s SPI value of the original IKE_SA. The message also has a Delete Payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.

Step 15: Judgment #7
The NUT transmits an INFORMATIONAL request. The message’s IKE_SA Initiator’s SPI value is the IKE_SA Initiator’s SPI value of the new IKE_SA initiated by the NUT at Step 9, and the message’s IKE_SA Responder’s SPI value is the IKE_SA Responder’s SPI value of the new IKE_SA initiated by the NUT at Step 9. The message also has a Delete Payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.

Step 18: Judgment #8
The NUT transmits an INFORMATIONAL response with no payload.

Possible Problems:

- Each NUT has the different lifetime of SA
- Step 13 (INFORMATIONAL request to delete the original IKE_SA) can possibly switch the place with Step 15 (INFORMATIONAL request to delete the new IKE_SA).
Test IKEv2.EN.I.1.2.6.6: Simultaneous IKE_SA Rekeying with retransmission

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4718] - Sections 5.11.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

### Packet #5: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>R (bit 5 of Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Message ID</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>E Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>42 (D)</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
<td></td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
<td></td>
</tr>
<tr>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
<td></td>
</tr>
<tr>
<td>D Payload</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1 (IKE_SA)</td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td># of SPIs</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Security Parameter Index</td>
<td>none</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. TN1 transmits a CREATE_CHILD_SA request to rekey IKE_SA to the NUT.
11. Observe the messages transmitted on Link A.
12. TN1 transmits an INFORMATONAL request to close the original IKE_SA. The message has a Delete Payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.
13. Observe the messages transmitted on Link A.
14. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.
Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request to rekey an IKE_SA. The message includes "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request has a SA payload including 1 (IKE) in the Protocol ID field, 8 in the SPI size field and new IKE_SA’s SPI value in the SPI field.

Step 11: Judgment #5
The NUT responds a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the proposal in the SA payload Response has a SA payload including 1 (IKE) in the Protocol ID field, 8 in the SPI size field and new IKE_SA’s responder’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT responds with an INFORMATIONAL response to the INFORMATIONAL request to close the original IKE_SA.

Step 14: Judgment #7
The NUT never retransmits a CREATE_CHILD_SA request transmitted at Step 9.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.I.1.2.6.7: Rekeying a CHILD_SA while Closing a CHILD_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.8: Closing a New CHILD_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.9: Rekeying a New CHILD_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.10: Rekeying an IKE_SA with half-open CHILD_SAs

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.11: Rekeying a CHILD_SA while rekeying an IKE_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.12: Rekeying an IKE_SA with half-closed CHILD_SAs

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.13: Closing a CHILD_SA while rekeying an IKE_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.2.6.14: Closing an IKE_SA while rekeying an IKE_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.ENG.1.2.6.15: Rekeying an IKE_SA while Closing an IKE_SA

This test case was deleted at revision 1.1.0.
Group 2.7. Non zero RESERVED fields

Test IKEv2.EN.I.1.2.7.1: Non zero RESERVED fields in CREATE_CHILD_SA response

Purpose:
To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:
- [RFC 4306] - Sections 2.5

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1 See Common Packet #2
Packet #2 See Common Packet #4
Packet #3 See Common Packet #19
Packet #4 See Common Packet #14
All RESERVED fields are set to one.

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link A.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT. All RESERVED fields in the message are set to one.
11. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 9: Judgment #4
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 11: Judgment #5
The NUT transmits an INFORMATIONAL request with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the inbound SPI value to be deleted as SPI.

Possible Problems:

- Each NUT has the different lifetime of SA.
Group 3. The INFORMATIONAL Exchange

Group 3.1. Header and Payload Formats

Test IKEv2.EN.I.1.3.1.1: Sending INFORMATIONAL Exchange

This test case was deleted at revision 1.1.0.
Group 3.2. Use of Retransmission Timers

Test IKEv2.EN.I.1.3.2.1: Retransmission of INFORMATIONAL request

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.I.1.3.2.2: Stop of retransmission of INFORMATIONAL request

This test case was deleted at revision 1.1.0.
Group 3.3. Non zero RESERVED fields

Test IKEv2.EN.I.1.3.3.1: Non zero RESERVED fields in INFORMATIONAL response

This test case was deleted at revision 1.1.0.
Group 3.4. Error Handling

Test IKEv2.EN.I.1.3.4.1: INVALID_SPI

This test case was deleted at revision 1.1.0.
Section 1.1.2. Endpoint to Security Gateway Tunnel

Group 1. The Initial Exchanges

Group 1.1. Header and Payload Formats

Test IKEv2.EN.I.2.1.1.1: Sending IKE_AUTH request

Purpose:
To verify an IKEv2 device transmits IKE_AUTH request using properly Header and Payloads format

References:
- [RFC 4306] - Sections 1.2, 2.15, 3.1, 3.2, 3.3, 3.5, 3.8, 3.10, 3.13 and 3.14

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

**Part A: IKE Header Format (ADVANCED)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.

**Part B: Encrypted Payload Format (ADVANCED)**
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_SA_INIT response to the NUT.
8. Observe the messages transmitted on Link A.

**Part C: IDi Payload Format (ADVANCED)**
9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. TN1 responds with an IKE_SA_INIT response to the NUT.
12. Observe the messages transmitted on Link A.

**Part D: AUTH Payload Format (ADVANCED)**
13. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. TN1 responds with an IKE_SA_INIT response to the NUT.
16. Observe the messages transmitted on Link A.

**Part E: SA Payload Format (ADVANCED)**
17. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. TN1 responds with an IKE_SA_INIT response to the NUT.
20. Observe the messages transmitted on Link A.

**Part F: TSi Payload Format (ADVANCED)**
21. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
22. Observe the messages transmitted on Link A.
23. TN1 responds with an IKE_SA_INIT response to the NUT.
24. Observe the messages transmitted on Link A.

**Part G: TSr Payload Format (ADVANCED)**
25. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. TN1 responds with an IKE_SA_INIT response to the NUT.
28. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted IKE Header containing following values:
An IKE_SA Initiator’s SPI field is set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.

An IKE_SA Responder’s SPI field is set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.

A Next Payload field is set to Encrypted Payload (46).

A Major Version field is set to 2.

A Minor Version field is set to zero.

An Exchange Type field is set to IKE_AUTH (35).

A Flags field is set to (00010000)₂ = (16)₁₀.

A Message ID field is set to 1.

A Length field is set to the length of the message (header + payloads) in octets.

**Part B**

**Step 6: Judgment #1**

The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 8: Judgment #2**

The NUT transmits an IKE_AUTH request including properly formatted Encrypted Payload containing following values:
• A Next Payload field is set to IDi Payload (35).
• A Critical field is set to zero.
• A RESERVED field is set to zero.
• A Payload Length field is set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Checksum Data.
• An Initialization Vector field is set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
• An Encrypted IKE Payloads field is set to subsequent payloads encrypted by ENCR_3DES.
• A Padding field is set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
• A Pad Length field is set to the length of the Padding field.
• An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

Part C

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted ID Payload containing following values:

Figure 37 ID Payload format

• A Next Payload field is set to AUTH Payload (39).
• A Critical field is set to zero.
• A RESERVED field is set to zero.
• A Payload Length field is set to length of the current payload. It is 24 bytes for ID_IPV6_ADDR.
• An ID Type field is set to ID_IPV6_ADDR (5).
• A RESERVED field is set to zero.
• An Identification Data field is set to the NUT address.

Part D

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted AUTH Payload containing following values:

- A Next Payload field is set to SA Payload (33).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 28 bytes for PRF_HMAC_SHA1.
- An Auth Method field is set to Shared Key Message Integrity Code (2).
- A RESERVED field is set to zero.
- An Authentication Data field is set to correct authentication value according to the manner described in RFC. It is 160 bytes length in PRF_HMAC_SHA1 case.

**Part E**

**Step 18: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field is set to TSi Payload (44).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.

The following proposal must be included in Proposals field.
IPv6 FORUM TECHNICAL DOCUMENT

Figure 41 Proposal sub-structure format

Transform field is set to following (There are 3 Transform Structures).

Proposal #1
- A 0 or 2 field is set to zero if this structure is the last proposal, otherwise set to 2.
- A RESREVD field is set to zero.
- A Proposal Length field is set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field is set to 1 if this structure is the first proposal, otherwise set to 1 greater than the previous proposal.
- A Protocol ID field is set to ESP (3).
- A SPI Size field is set to 4.
- A # of Transforms field is set to 3.
- A SPI field is set to the sending entity’s SPI (4 octets value)

Transform field is set to following (There are 3 Transform Structures).

Figure 42 Transform sub-structure format

Transform #1
- A 0 or 3 field is set to zero if this structure is the last proposal, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field is set to ENCR (1).
- A RESERVED field is set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
• A 0 or 3 field is set to zero if this structure is the last proposal, otherwise set to 3.
• A RESERVED field is set to zero.
• A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
• A Transform Type field is set to INTEG (3).
• A RESERVED field is set to zero.
• A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3
• A 0 or 3 field is set to zero if this structure is the last proposal, otherwise set to 3.
• A RESERVED field is set to zero.
• A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
• A Transform Type field is set to ESN (5).
• A RESERVED field is set to zero.
• A Transform ID set to No Extended Sequence Numbers (0).

Part F

Step 22: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 24: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted TSi Payload containing following values:

![Figure 43 TSi Payload format](image)

• A Next Payload field is set to TSr Payload (45).
• A Critical field is set to zero.
• A RESERVED field is set to zero.
• A Payload Length field is set to length of the current payload.
• A Number of TSs field is set to the number of actual traffic selectors.
• A RESERVED field is set to zero.

The following traffic selector must be included in Traffic Selectors field.
Figure 44 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to less than or equal to NUT address.
- A Ending Address field is set to greater than or equal to NUT address.

Part G

Step 26: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 28: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted TSr Payload containing following values:

Figure 45 TSr Payload format

- A Next Payload field is set to zero.
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Number of TSs field is set to the number of actual traffic selectors.
- A RESERVED field is set to zero.
The following traffic selector must be included in Traffic Selectors field.

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to less than or equal to Prefix Y.
- An Ending Address field is set to less than or equal to Prefix Y.

Possible Problems:

- IKE_AUTH request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- The implementation may not set single proposal by the implementation policy. In this case, Security Association Payload contains multiple proposals.

- The implementation may not set single traffic selector by the implementation policy. In this case, Traffic Selector Payload contains multiple proposals.
• Each of transforms can be located in the any order.
Test IKEv2.EN.I.2.1.1.2: Use of CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 1.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #20</td>
</tr>
</tbody>
</table>

Part A (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH1 transmits an Echo Request and TN1 forwards an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Possible Problems:

- None.
Group 1.2. Requesting an Internal Address on a Remote Network

Test IKEv2.EN.I.2.1.2.1: Sending CFG_REQUEST

Purpose:
To verify an IKEv2 device transmits IKE_AUTH request using properly configuration payload format.

References:
- [RFC 4306] - Sections 3.15

Test Setup:
- Network Topology
  Connect the devices according to the following topology.

  ![Network Topology Diagram]

  Prefix A::any_interface_ID (External Address)
  Prefix Y::1 (Internal Address) (assigned by CP)

  Link A (Prefix A, MTU=1500)
  Prefix A = 2001:0db8:0001:0001::/64
  Prefix X = 2001:0db8:000f:0001::/64
  Prefix Y = 2001:0db8:000f:0002::/64

  Link X (Prefix X, MTU=1500)
  Prefix X::1

  Link Y (Prefix Y, MTU=1500)
  Prefix Y::f

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REQUEST for
INTERNAL_IP6_ADDRESS. The traffic selector must be configured by the following table.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source Address Range</th>
<th>Next Layer Protocol</th>
<th>Port Range</th>
<th>Destination Address Range</th>
<th>Next Layer Protocol</th>
<th>Port Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbound</td>
<td>Link Y</td>
<td>ANY</td>
<td>ANY</td>
<td>NUT (internal address)</td>
<td>ANY</td>
<td>ANY</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT (internal address)</td>
<td>ANY</td>
<td>ANY</td>
<td>Link Y</td>
<td>ANY</td>
<td>ANY</td>
</tr>
</tbody>
</table>

* NUT must propose Traffic Selector covering above address range.

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 responds with an IKE_SA_INIT response to the NUT.
6. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted Configuration Payload containing following values:
Figure 47 Configuration Payload format

- A Next Payload field is set to SA Payload (33).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A CFG Type field is set to CFG_REQUEST (1).
- A RESERVED field is set to zero.

The following configuration attribute must be included in Configuration Attributes field.

Figure 48 Configuration Attributes format

Configuration Attribute #1
- Reserved field is set to zero.
- Attribute Type field is set to INTERNAL_IP6_ADDRESS (8).
- Length field is set to zero.
- Value field is empty.

Possible Problems:

- The implementation may not set single configuration attribute by the implementation policy. In this case, Configuration Payload contains multiple configuration attributes.
Test IKEv2.EN.I.2.1.2.2: Receipt of CFG_REPLY

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 2.19 and 3.15

Test Setup:

- Network Topology
  Connect the devices according to the following topology.

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REQUEST for INTERNAL_IP6_ADDRESS. The traffic selector must be configured by the following table.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
</table>

- Prefix A::any_interface_ID (External Address)
- Prefix Y::1 (Internal Address) (assigned by CP)
- Link A (Prefix A, MTU=1500)
- Link X (Prefix X, MTU=1500)
- Link Y (Prefix Y, MTU=1500)
### IPv6 FORUM TECHNICAL DOCUMENT

#### IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Address Range</th>
<th>Next Layer Protocol</th>
<th>Port Range</th>
<th>Address Range</th>
<th>Next Layer Protocol</th>
<th>Port Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbound</td>
<td>Link Y</td>
<td>ANY</td>
<td>NUT (internal address)</td>
<td>ANY</td>
<td>ANY</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT (internal address)</td>
<td>ANY</td>
<td>Link Y</td>
<td>ANY</td>
<td>ANY</td>
</tr>
</tbody>
</table>

* NUT must propose Traffic Selector covering above address range.

- **Pre-Sequence and Cleanup Sequence**
  - IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Below</td>
</tr>
</tbody>
</table>

- **Packet #2: IKE_AUTH response packet**

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Next Payload 47 (CP)</td>
</tr>
<tr>
<td></td>
<td>Other fields are same as Common Packet #6</td>
</tr>
<tr>
<td>CP Payload</td>
<td>Next Payload 33 (SA)</td>
</tr>
<tr>
<td></td>
<td>Critical 0</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Payload Length 29</td>
</tr>
<tr>
<td></td>
<td>CFG Type 2 (CFG_REPLY)</td>
</tr>
<tr>
<td></td>
<td>RESERVED 0</td>
</tr>
<tr>
<td></td>
<td>Configuration Attributes See below</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as Common Packet #6</td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as Common Packet #6</td>
</tr>
</tbody>
</table>

| Configuration Attributes | Reserved | 0 |
Packet #3: Echo Request packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #20</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESP</td>
<td>Same as Common Packet #20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Prefix Y-1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>Prefix Y-1</td>
</tr>
</tbody>
</table>

| ICMPv6 Header | Same as Common Packet #20 |

**Part A (ADVANCED)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH1 transmits an Echo Request to NUT internal address and TN1 forwards an Echo Request with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96. The inner packet is sent from NUT internal address.

**Possible Problems:**

- The implementation may not set single configuration attribute by the implementation policy. In this case, Configuration Payload contains multiple configuration attributes.
Test IKEv2.EN.I.2.1.2.3: Non zero RESERVED fields in Configuration Payload

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the following topology.

<table>
<thead>
<tr>
<th>Prefix A: any_interface_ID (External Address)</th>
<th>Prefix Y::1 (Internal Address) (assigned by CP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link A (Prefix A, MTU=1500)</td>
<td></td>
</tr>
<tr>
<td>TR1 (Router)</td>
<td></td>
</tr>
<tr>
<td>Link X (Prefix X, MTU=1500)</td>
<td></td>
</tr>
<tr>
<td>TN1 (SGW)</td>
<td></td>
</tr>
<tr>
<td>Link Y (Prefix Y, MTU=1500)</td>
<td></td>
</tr>
<tr>
<td>TH1 (Host)</td>
<td></td>
</tr>
</tbody>
</table>

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REQUEST for INTERNAL_IP6_ADDRESS. The traffic selector must be configured by the following table.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

### IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Address Range</th>
<th>Next Layer Protocol</th>
<th>Port Range</th>
<th>Address Range</th>
<th>Next Layer Protocol</th>
<th>Port Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbound</td>
<td>Link Y</td>
<td>ANY</td>
<td>ANY</td>
<td>NUT (internal address)</td>
<td>ANY</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT (internal address)</td>
<td>ANY</td>
<td>Link Y</td>
<td>ANY</td>
<td>ANY</td>
</tr>
</tbody>
</table>

* NUT must propose Traffic Selector covering above address range.

- **Pre-Sequence and Cleanup Sequence**
  - IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Below</td>
</tr>
</tbody>
</table>

- **Packet #2: IKE_AUTH response packet**

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Next Payload 47 (CP)</td>
</tr>
</tbody>
</table>

Other fields are same as Common Packet #6

<table>
<thead>
<tr>
<th>CP Payload</th>
<th>Next Payload 33 (SA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>1</td>
</tr>
<tr>
<td>Payload Length</td>
<td>29</td>
</tr>
<tr>
<td>CFG Type</td>
<td>2 (CFG_REPLY)</td>
</tr>
<tr>
<td>RESERVED</td>
<td>1</td>
</tr>
<tr>
<td>Configuration Attributes</td>
<td>See below</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Same as Common Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as Common Packet #6</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as Common Packet #6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Configuration Attributes</th>
<th>Reserved</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Type</td>
<td>INTERNAL_IP6_ADDRESS</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>Value</td>
<td>IPv6 address</td>
<td></td>
</tr>
<tr>
<td>Prefix-length</td>
<td>128</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y:1</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix Y:1</td>
<td></td>
</tr>
</tbody>
</table>

- Packet #3: Echo Request packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #20</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESP</td>
<td>Same as Common Packet #20</td>
</tr>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
</tr>
<tr>
<td></td>
<td>Prefix Y:f</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Destination Address</td>
</tr>
<tr>
<td></td>
<td>Prefix Y:1</td>
</tr>
</tbody>
</table>

**Part A (ADVANCED)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH1 transmits an Echo Request to NUT internal address and TN1 forwards an Echo Request with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES","AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96. The inner packet is sent from NUT internal address.

**Possible Problems:**
- The implementation may not set single configuration attribute by the implementation policy. In this case, Configuration Payload contains multiple configuration attributes.
Test IKEv2.EN.I.2.1.2.4: Receipt of IKE_AUTH response without CFG_REPLY

Purpose:
To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:
- [RFC 4718] - Sections 6.8

Test Setup:
- Network Topology
  Connect the devices according to the following topology.

```
+-------------------+            +-------------------+            +-------------------+
|                  |            |                  |            |                  |
|                  |            |                  |            |                  |
|                  |            |                  |            |                  |
|                  |            |                  |            |                  |
NUT (End-Node)    |            |                  |            |                  |
+-------------------+            +-------------------+            +-------------------+
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
Prefix A::any_interface_ID (External Address)
Prefix Y::1 (Internal Address) (assigned by CP)

    Prefix A = 2001:0db8:0001:0001::/64
    Prefix X = 2001:0db8:000f:0001::/64
    Prefix Y = 2001:0db8:000f:0002::/64

+-------------------+            +-------------------+            +-------------------+
|                  |            | Pattern: fe80:f    |            |                  |
|                  |            | TR1 (Router)       |            |                  |
+-------------------+            +-------------------+            +-------------------+
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
Prefix X::1          |            |                      |            |                      |
+-------------------+            +-------------------+            +-------------------+
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
TN1 (SGW)            |            |                      |            |                      |
+-------------------+            +-------------------+            +-------------------+
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
Prefix Y::f           |            |                      |            |                      |
+-------------------+            +-------------------+            +-------------------+
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
 |                      |            |                      |            |                      |
TH1 (Host)            |            |                      |            |                      |
```

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REQUEST for INTERNAL_IP6_ADDRESS. The traffic selector must be configured by the following table.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inbound</td>
<td>Outbound</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>Link Y</td>
<td>NUT (internal address)</td>
<td></td>
</tr>
</tbody>
</table>

* NUT must propose Traffic Selector covering above address range.

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

**Procedure:**

- Packet #1: See Common Packet #2
- Packet #2: See Below
- Packet #3: See Common Packet #17

**Packet #2: IKE_AUTH response packet**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>Idr Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Next Payload 33 (SA)</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as Common Packet #6</td>
</tr>
<tr>
<td>TSR Payload</td>
<td>See below</td>
</tr>
</tbody>
</table>

**Traffic Selector**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y:1</td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix Y:1</td>
</tr>
</tbody>
</table>

*Part A (ADVANCED)*

IPv6 FORUM TECHNICAL DOCUMENT 320 IPv6 Ready Logo Program IKEv2
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT. The message does not include any Configuration payloads.
6. TH1 transmits an INFORMATIONAL request with no payload to NUT.
7. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an INFORMATIONAL response with no payload to the TN1.

Possible Problems:

- The implementation may not set single configuration attribute by the implementation policy. In this case, Configuration Payload contains multiple configuration attributes.
Test IKEv2.EN.I.2.1.2.5: Receipt of unrecognized Configuration Attributes

Purpose:

To verify an IKEv2 device properly handles unrecognized Configuration Attributes.

References:

- [RFC 4306] - Sections 2.19 and 3.15

Test Setup:

- Network Topology
  Connect the devices according to the following topology.

  ![Network Topology Diagram]

  Prefix A::any_interface_ID (External Address)
  Prefix Y::1 (Internal Address) (assigned by CP)

  Link A (Prefix A, MTU=1500)

  Prefix A = 2001:0db8:0001:0001::/64
  Prefix X = 2001:0db8:000f:0001::/64
  Prefix Y = 2001:0db8:000f:0002::/64

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REQUEST for INTERNAL_IP6_ADDRESS. The traffic selector must be configured by the following table.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

### Pre-Sequence and Cleanup Sequence
IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

- **Packet #2: IKE_AUTH response packet**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>Idr Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Next Payload 47 (CP)</td>
</tr>
<tr>
<td></td>
<td>Other fields are same as Common Packet #6</td>
</tr>
<tr>
<td>CP Payload</td>
<td>Next Payload 33 (SA)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>29</td>
</tr>
<tr>
<td>CFG Type</td>
<td>2 (CFG_REPLY)</td>
</tr>
<tr>
<td>RESERVED</td>
<td>0</td>
</tr>
<tr>
<td>Configuration Attributes</td>
<td>See below</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as Common Packet #6</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>Configuration Attributes</td>
<td>Reserved 0</td>
</tr>
<tr>
<td>Attribute Type</td>
<td>32767</td>
</tr>
</tbody>
</table>

* NUT must propose Traffic Selector covering above address range.
### IPv6 FORUM TECHNICAL DOCUMENT

#### IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y::1</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix Y::1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Length</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>IPv6 address Prefix Y::1</td>
</tr>
<tr>
<td>Prefix-length</td>
<td>128</td>
</tr>
</tbody>
</table>

**Part A (ADVANCED)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT. The message includes a Configuration Attribute of unrecognized Attribute Type.
6. TH1 transmits an INFORMATIONAL request with no payload to NUT.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an INFORMATIONAL response with no payload to the TN1.

**Possible Problems:**

- The implementation may not set single configuration attribute by the implementation policy. In this case, Configuration Payload contains multiple configuration attributes.
Section 1.2. Responder

Section 1.2.1. Endpoint-to-Endpoint Transport

Group 1. The Initial Exchanges
Group 1.1. Header and Payload Formats

Test IKEv2.EN.R.1.1.1.1: Sending IKE_SA_INIT response

Purpose:

To verify an IKEv2 device transmits an IKE_SA_INIT response using properly Header and Payloads format

References:

- [RFC4306] - Section 1.2, 2.10, 3.1, 3.2, 3.3, 3.4 and 3.9
- [RFC 4718] - Sections 7.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
</table>

Part A: IKE Header Format (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Part B: SA Payload Format (BASIC)
3. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
4. Observe the messages transmitted on Link A.

Part C: KE Payload Format (BASIC)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.

Part D: Nonce Payload Format (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including properly formatted IKE Header containing following values:

![Figure 49 Header format](image)

- An IKE_SA Initiator’s SPI field is set to IKE_SA Initiator’s SPI field value supplied in the first IKE_SA_INIT request message.
- An IKE_SA Responder’s SPI field is set to a 64-bits value chosen by the NUT. It MUST not be zero.
- A Next Payload field is set to SA Payload (33).
- A Major Version field is set to 2.
- A Minor Version field is set to zero.
- An Exchange Type field is set to IKE_SA_INIT (34).
- A Flags field is set to (00000100)₂ = (4)₁₀.
- A Message ID field is set to zero.
- A Length field is set to the length of the message (header + payloads) in octets.

Part B

Step 4: Judgment #1
Figure 50 SA Payload contents

The NUT transmits an IKE_SA_INIT response including properly formatted SA Payload containing following values (refer following figures):

Figure 51 SA Payload format

- A Next Payload field is set to KE Payload (34).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.

The following proposal must be included in Proposals field.
Proposal #1
- A 0 or 2 field is set to zero if this structure is the last proposal, otherwise set to 2.
- A RESREVD field is set to zero.
- A Proposal Length field is set to length of this proposal, including all transforms and attributes. It is 40 bytes for this proposal according to Common Configuration.
- A Proposal # field is set to 1.
- A Protocol ID field is set to IKE (1).
- A SPI Size field is set to zero.
- A # of Transforms field is set to 4.

A Transform field is set to following (There are 4 Transform Structures).

Transform #1
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field is set to ENCR (1).
- A RESERVED field is set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
IPv6 FORUM TECHNICAL DOCUMENT

- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for PRF_HMAC_SHA1.
- A Transform Type field is set to PRF (2).
- A RESERVED field is set to zero.
- A Transform ID set to PRF_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field is set to INTEG (3).
- A RESERVED field is set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #4
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for 1024 MODP Group.
- A Transform Type field is set to D-H (4).
- A RESERVED field is set to zero.
- A Transform ID set to Group2 (2).

Part C
Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including properly formatted KE Payload containing following values:

- A Next Payload field is set to Nonce Payload (40).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 136 bytes for Group 2.
- A DH Group field is set to Group2 (2).
- A RESERVED field is set to zero.
- A Key Exchange Data field is set to Diffie-Hellman public value. The length of
the Key Exchange Data field must be equal to 1024bit.
- The length of the Key Exchange Data field must be equal to 1024bit.

**Part D**

**Step 8: Judgment #4**
The NUT transmits an IKE_SA_INIT response including properly formatted Nonce Payload containing following values:

- A Next Payload field is set to zero.
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Nonce Data field is set to random data generated by the transmitting entity.
- The size of the Nonce must between 16 and 256 octets.

**Possible Problems:**

- IKE_SA_INIT response has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- Each of transforms can be located in the any order.
Test IKEv2.EN.R.1.1.1.2: Sending IKE_AUTH response

Purpose:
To verify an IKEv2 device transmits an IKE_AUTH response using properly Header and Payloads format

References:
- [RFC 4306] - Sections 1.2, 2.15, 3.1, 3.2, 3.3, 3.5, 3.8, 3.10, 3.13 and 3.14

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
</tbody>
</table>

Part A: IKE Header Format (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.

Part B: Encrypted Payload Format (BASIC)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
8. Observe the messages transmitted on Link A.
Part C: IDr Payload Format (BASIC)
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
12. Observe the messages transmitted on Link A.

Part D: AUTH Payload Format (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.

Part E: Notify Payload Format (BASIC)
17. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
20. Observe the messages transmitted on Link A.

Part F: SA Payload Format (BASIC)
21. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
24. Observe the messages transmitted on Link A.

Part G: TSi Payload Format (BASIC)
25. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
28. Observe the messages transmitted on Link A.

Part H: TSr Payload Format (BASIC)
29. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
30. Observe the messages transmitted on Link A.
31. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
32. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted IKE Header containing following values:

![Header format](image)

- An IKE_SA_Initiator’s SPI field is set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA_Responder’s SPI field is set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field is set to Encrypted Payload (46).
- A Major Version field is set to 2.
- A Minor Version field is set to zero.
- An Exchange_Type field is set to IKE_AUTH (35).
- A Flags field is set to (00000100)₂ = (4)₁₀.
- A Message_ID field is set to 1.
- A Length field is set to the length of the message (header + payloads) in octets.

**Part B**

**Step 6: Judgment #1**

The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 8: Judgment #2**

The NUT transmits an IKE_AUTH response including properly formatted Encrypted Payload containing following values:
Figure 57 Encrypted payload

- A Next Payload field is set to IDr Payload (36).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Checksum Data.
- An Initialization Vector field is set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
- An Encrypted IKE Payloads field is set to subsequent payloads encrypted by ENCR_3DES.
- A Padding field is set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
- A Pad Length field is set to the length of the Padding field.
- An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

Part C

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted ID Payload containing following values:
IPv6 FORUM TECHNICAL DOCUMENT

Figure 58 ID Payload format

- A Next Payload field is set to AUTH Payload (39).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 24 bytes for ID_IPV6_ADDR.
- An ID Type field is set to ID_IPV6_ADDR (5).
- A RESERVED field is set to zero.
- An Identification Data field is set to the NUT address.

Part D

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted AUTH Payload containing following values:

Figure 59 AUTH Payload format

- A Next Payload field is set to Notify Payload (41).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 28 bytes for PRF_HMAC_SHA1
- An Auth Method field is set to Shared Key Message Integrity Code (2).
- A RESERVED field is set to zero.
- An Authentication Data field is set to correct authentication value.
Part E

Step 18: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 20: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted Notify Payload containing following values:

Figure 60 Notify Payload format

1. A Next Payload field is set to SA Payload (33).
2. A Critical field is set to zero.
3. A RESERVED field is set to zero.
4. A Payload Length field is set to length of the current payload. It is 8 bytes for USE_TRANSPORT.
5. A Protocol ID field is set to IKE_SA (1).
6. A SPI Size field is set to zero.
7. A Notify Message Type field is set to USE_TRANSPORT_MODE (16391)

Part F

Step 22: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 24: Judgment #2
Figure 61 SA Payload contents

The NUT transmits an IKE_AUTH response including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field is set to TSi Payload (44).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.

The following proposal must be included in Proposals field.
Proposal #1
- A 0 or 2 field is set to zero if this structure is the last proposal, otherwise set to 2.
- A RESERVD field is set to zero.
- A Proposal Length field is set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field is set to 1.
- A Protocol ID field is set to ESP (3).
- A SPI Size field is set to 4.
- A # of Transforms field is set to 3.
- A SPI field is set to the sending entity’s SPI (4 octets value)

Transform field is set to following (There are 3 Transform Structures).

Transform #1
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field is set to ENCR (1).
- A RESERVED field is set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
IPv6 FORUM TECHNICAL DOCUMENT

Part G

Step 26: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 28: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted TSi Payload containing following values:

Figure 65 TSi Payload format

- A Next Payload field is set to TSr Payload (45).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Number of TSs field is set to 1.
- A RESERVED field is set to zero.

The following traffic selector must be included in Traffic Selectors field.
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 READY LOGO PROGRAM IKEv2

Figure 66 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to TN1 address.
- An Ending Address field is set to TN1 address.

Part H

Step 30: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 32: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted TSr Payload containing following values:

Figure 67 TSr Payload format

- A Next Payload field is set to zero.
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Number of TSs field is set to 1.
- A RESERVED field is set to zero.
Traffic Selectors field is set to following.

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to NUT address.
- An Ending Address field is set to NUT address.

Possible Problems:

- IKE_AUTH response has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- Each of transforms can be located in the any order.
IPv6 FORUM TECHNICAL DOCUMENT

Test IKEv2.EN.R.1.1.1.3: Use of CHILD_SA

Purpose:

To verify an IKEv2 device properly handles CHILD_SA negotiated by the Initial Exchanges using Pre-shared key.

References:

- [RFC 4306] - Sections 1.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Possible Problems:**

- None.
Group 1.2. Use of Retransmission Timers

Test IKEv2.EN.R.1.1.2.1: Receipt of retransmitted IKE_SAInicial request

Purpose:
To verify an IKEv2 device transmits an IKE_SAInicial response when the device received a retransmitted IKE_SAInicial request.

References:
- [RFC 4306] - Sections 2.1, 2.2 and 2.4
- [RFC 4718] - Sections 2.2 and 2.3

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #1</td>
</tr>
<tr>
<td></td>
<td>(The Message ID is the same as Packet #1)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN starts to negotiate with TN1 by sending IKE_SAInicial request.
2. Observe the messages transmitted on Link A.
3. Observe the messages transmitted on Link A.
4. TN1 retransmits same IKE_SAInicial request as the message transmitted in Step 1 to the
5. Observe the messages transmitted on Link A.

**Observable Results:**

*Part A*

**Step 2: Judgment #1**
The NUTtransmits an IKE_SA_INIT response including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 3: Judgment #2**
The NUT never retransmits the same IKE_SA_INIT response as the response transmitted at Step 2.

**Step 5: Judgment #3**
The NUT transmits the same IKE_SA_INIT response as the response transmitted at Step 2.

**Possible Problems:**

- None.
Test IKEv2.EN.R.1.1.2.2: Receipt of retransmitted IKE_AUTH request

Purpose:

To verify an IKEv2 device transmits an IKE_AUTH response when the device received a retransmitted IKE_AUTH request.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td></td>
<td>(The Message ID is the same as Packet #1)</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of an IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. Observe the messages transmitted on Link A.
6. TN1 retransmits the same IKE_AUTH request as the request transmitted in Step 3 to the NUT.
7. Observe the messages transmitted on Link A.

**Observable Results:**

*Part A*

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 5: Judgment #3**
The NUT never retransmits the same IKE_AUTH response as the response transmitted at Step 4.

**Step 7: Judgment #4**
The NUT transmits the same IKE_AUTH response as the response transmitted at Step 4.

**Possible Problems:**

- None.
Group 1.3. State Synchronization and Connection Timeouts

Test IKEv2.EN.R.1.1.3.1: State Synchronization with ICMP messages

Purpose:

To verify that an IKEv2 device doesn't conclude that the other endpoint has failed by receiving ICMP Error messages.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet 1</td>
<td></td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 349 IPv6 Ready Logo Program IKEv2
Packet #4: ICMPv6 Destination Unreachable

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TR1's Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT's Global Address on Link A</td>
</tr>
<tr>
<td>IPv6 Header</td>
<td>Type</td>
<td>1</td>
</tr>
<tr>
<td>Code</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. After reception of an Echo Reply from NUT, TR1 transmits ICMP Destination Unreachable Message to the NUT.
8. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
9. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**

The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**

The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**

The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Possible Problems:**

- None.
Test IKEv2.EN.R.1.1.3.2: State Synchronization with IKE messages

Purpose:

To verify that an IKEv2 device doesn't conclude that the other endpoint has failed by receiving cryptographically unprotected IKE message.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

```
Packet #1 See Common Packet #1
Packet #2 See Common Packet #3
Packet #3 See Common Packet #19
Packet #4 See below
```
Packet #4: cryptographically unprotected INFORMATIONAL request

<table>
<thead>
<tr>
<th>IP/UDP/IKEv2 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
<th>Source Port</th>
<th>Destination Port</th>
<th>IKE_SA Initiator’s SPI</th>
<th>IKE_SA Responder’s SPI</th>
<th>Next Payload</th>
<th>Major Version</th>
<th>Minor Version</th>
<th>Exchange Type</th>
<th>I (3)</th>
<th>X (0)</th>
<th>V (4)</th>
<th>R (5)</th>
<th>X (6-7)</th>
<th>Message ID</th>
<th>Length</th>
<th>Notify Message Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>TN1’s Global Address on Link A</td>
<td>NUT’s Global Address on Link X</td>
<td>500</td>
<td>500</td>
<td>any</td>
<td>any</td>
<td>41 (N)</td>
<td>2</td>
<td>0</td>
<td>37 (INFORMATIONAL)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>any</td>
<td>any</td>
<td>11 (INVALID_SPI)</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>Destination Port</td>
<td>500</td>
<td>500</td>
<td>any</td>
<td>any</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>IKE_SA Responder’s SPI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
<td>Critical</td>
<td>Reserved</td>
<td>Payload Length</td>
<td>Protocol ID</td>
<td>SPI Size</td>
<td>Notify Message Type</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>3 (ESP)</td>
<td>0</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. After reception of an Echo Reply from NUT, TN1 transmits a cryptographically unprotected INFORMATIONAL request with Notify payload of type INVALID_SPI to the NUT.
8. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Possible Problems:**

- None
Test IKEv2.EN.R.1.1.3.3: Close connections when receiving INITIAL_CONTACT

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.R.1.1.3.4: Receiving Liveness check

Purpose:

To verify that an IKEv2 device can respond to INFORMATIONAL request for liveness check.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an
6. Observe the messages transmitted on Link A.

**Observable Results:**

*Part A*

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits an INFOMATIONAL Response followed by an Encrypted payload with no payloads contained in it.

**Possible Problems:**

- None
Test IKEv2.EN.R.1.1.3.5: Receiving Delete Payload for IKE_SA

Purpose:

To verify an IKEv2 device can respond to INFORMATIONAL request with a Delete Payload, when IKE_SA is deleted.

References:

- [RFC 4306] - Sections 2.4 and 3.11

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

- Packet #3: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td></td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>Destination Port</td>
<td></td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP Header</th>
<th>Source Port</th>
<th>Destination Port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500</td>
<td>500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IKEv2 Header</th>
<th>IKEv2 SA Initiator’s SPI</th>
<th>IKEv2 SA Responder’s SPI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>any</td>
<td>any</td>
</tr>
</tbody>
</table>
Next Payload 46 (E)  
Major Version 2  
Minor Version 0  
Exchange Type 37 (INFORMATIONAL)  
X (bits 0-2 of Flags) 0  
I (bit 3 of Flags) any  
V (bit 4 of Flags) 0  
R (bit 5 of Flags) 0  
X (bits 6-7 Flags) 0  
Message ID 2  
Length any  

E Payload  
Next Payload 42 (D)  
Critical 0  
Reserved 0  
Payload Length any  
Initialization Vector The same value as block length of the underlying encryption algorithm  
Encrypted IKE Payloads Subsequent payloads encrypted by underlying encryption algorithm  
Padding Any value which to be a multiple of the encryption block size  
Pad Length The length of the Padding field  
Integrity Checksum Data The Cryptographic checksum of the entire message  

D Payload  
Next Payload 0  
Critical 0  
Reserved 0  
Payload Length 8  
Protocol ID 1 (IKE_SA)  
SPI Size 0  
# of SPIs 0  
Security Parameter Index none  

Part A: (BASIC)  
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.  
2. Observe the messages transmitted on Link A.  
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.  
4. Observe the messages transmitted on Link A.  
5. TN1 transmits an INFORMATIONAL request with a Delete payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.  
6. Observe the messages transmitted on Link A.  

Observable Results:  

Step 2: Judgment #1  
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.  

Step 4: Judgment #2  
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.  

Step 7: Judgment #3  
The NUT transmits an INFORMATIONAL response with no payloads.  

Possible Problems:
• None
Test IKEv2.EN.R.1.1.3.6: Receiving Delete Payload for CHILD_SA

Purpose:
To verify an IKEv2 device can respond to INFORMATIONAL request with a Delete Payload, when CHILD_SAs are deleted.

References:
- [RFC 4306] - Sections 2.4 and 3.11

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

- Packet #3: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TN1’s Global Address on Link X</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port 500</td>
<td>Destination Port 500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>IKE_SA Responder’s SPI</td>
</tr>
</tbody>
</table>
### IPv6 Ready Logo Program IKEv2

#### Next Payload 46 (E)
- Major Version: 2
- Minor Version: 0
- Exchange Type: 37 (INFORMATIONAL)
- X (bits 0-2 of Flags): 0
- I (bit 3 of Flags): any
- V (bit 4 of Flags): 0
- R (bit 5 of Flags): 0
- X (bits 6-7 Flags): 0
- Message ID: 2
- Length: any

#### Next Payload 42 (D)
- Critical: 0
- Reserved: 0
- Payload Length: any
- Initialization Vector: The same value as block length of the underlying encryption algorithm
- Encrypted IKE Payloads: Subsequent payloads encrypted by underlying encryption algorithm
- Padding: Any value which to be a multiple of the encryption block size
- Pad Length: The length of the Padding field
- Integrity Checksum Data: The Cryptographic checksum of the entire message

#### Next Payload 0
- Critical: 0
- Reserved: 0
- Payload Length: 12
- Protocol ID: 3 (ESP)
- SPI Size: 4
- # of SPIs: 1
- Security Parameter Index: NUT’s inbound CHILD_SA SPI value to be deleted

---

### Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an INFORMATIONAL request with a Delete payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the TN1’s inbound SPI value to be deleted as SPI value.
6. Observe the messages transmitted on Link A.

### Observable Results:

#### Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 7: Judgment #3**
The NUT transmits an INFORMATIONAL response with a Delete payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the NUT’s inbound SPI value to be deleted as SPI value.

### Possible Problems:
None
Group 1.4. Version Numbers and Forward Compatibility

Test IKEv2.EN.R.1.1.4.1: Receipt of a higher minor version number

Purpose:

To verify an IKEv2 device accepts a request with a higher minor version number and respond to the request.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the Common Packet #1</td>
</tr>
<tr>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td>Minor Version</td>
<td>1</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>KE Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
</tbody>
</table>

Packet #1: IKE_SA_INIT request

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request with a higher minor version number.
2. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.4.2: Receipt of a higher major version number

Purpose:
To verify an IKEv2 device drops a request with a higher major version number and send a notification message.

References:
- [RFC 4306] - Sections 2.5

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See below</th>
</tr>
</thead>
</table>

Packet#1:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the Common Packet #1</td>
</tr>
<tr>
<td>Major Version</td>
<td>3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>KE Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>Ni Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response with a Notify payload of type INVALID_MAJOR_VERSION containing following values:
A Next Payload field is set to zero.
A Critical field is set to zero.
A RESERVED field is set to zero.
A Payload Length field is set to length of the current payload.
A SPI Size field is set to zero.
A Notify Message Type field is set to INVALID_MAJOR_VERSION (5).
A Notification Data field is set to the highest version number it supports (2).

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.4.3: Unrecognized payload types and critical bit is not set

Purpose:
To verify an IKEv2 device ignores invalid payload types when the invalid type payload’s critical bit is not set.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

- Packet #1: See Common Packet #1
- Packet #2: See Common Packet #3
- Packet #3: See below

<table>
<thead>
<tr>
<th>Packet #3: CREATE_CHILD_SA request</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header                        All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>UDP Header                         All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>IKEv2 Header                       All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>E Payload                          Next Payload</td>
</tr>
<tr>
<td>Invalid Payload</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>N Payload</td>
</tr>
<tr>
<td>N Payload</td>
</tr>
<tr>
<td>SA Payload</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
</tr>
<tr>
<td>Tsi Payload</td>
</tr>
<tr>
<td>Tsr Payload</td>
</tr>
</tbody>
</table>

Other fields are same as Common Packet #13

**Part A: Invalid payload type 1 (BASIC)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 1 and the invalid payload’s critical flag is not set. The request includes a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

**Part B: Invalid payload type 32 (BASIC)**
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 32 and the invalid payload’s critical flag is not set. The request includes a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

**Part C: Invalid payload type 49 (BASIC)**
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 49 and the invalid payload’s critical flag is not set. The request includes a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

**Part D: Invalid payload type 255 (BASIC)**
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT.
type to the NUT. The E payload’s IKE Header Next Payload field is set to 255 and the invalid payload’s critical flag is not set. The request includes a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.

24. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part B

Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 10: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 12: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part C

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 18: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part D

Step 20: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 22: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 24: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- None.
Test IKEv2.EN.R.1.1.4.4: Unrecognized payload types and critical bit is set

Purpose:

To verify an IKEv2 device drops invalid payload types when the invalid type payload’s critical bit is set.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

- Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>All fields are same as Common Packet #13 Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>E Payload</td>
<td>All fields are same as Common Packet #13 Payload</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

### Part A: Invalid payload type 1 and Critical bit is set (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The CREATE_CHILD_SA request’s IKE Header Next Payload field is set to 1 and the pointed payload’s Critical bit is set.
6. Observe the messages transmitted on Link A.

### Part B: Invalid payload type 32 and Critical bit is set (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The CREATE_CHILD_SA request’s IKE Header Next Payload field is set to 32 and the pointed payload’s Critical bit is set.
12. Observe the messages transmitted on Link A.

### Part C: Invalid payload type 49 and Critical bit is set (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. After reception of IKE_AUTH response from the NUT, TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The CREATE_CHILD_SA request’s IKE Header Next Payload field is set to 49 and the pointed payload’s Critical bit is set.
18. Observe the messages transmitted on Link A.

### Part D: Invalid payload type 255 and Critical bit is set (BASIC)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH response from the NUT, TN1 transmits a
CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The CREATE_CHILD_SA request’s IKE Header Next Payload field is set to 255 and the pointed payload’s Critical bit is set.

24. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response. The response has a Notify payload of type UNSUPPORTED_CRITICAL_PAYLOAD with the invalid payload type value (1).

**Part B**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 10: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 12: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response. The response has a Notify payload of type UNSUPPORTED_CRITICAL_PAYLOAD with the invalid payload type value (32).

**Part C**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response. The response has a Notify payload of type UNSUPPORTED_CRITICAL_PAYLOAD with the invalid payload type value (49).

**Part D**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 22: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 24: Judgment #3
The NUT transmits a CREATE_CHILD_SA response. The response has a Notify payload of type UNSUPPORTED_CRITICAL_PAYLOAD with the invalid payload type value (255).

Possible Problems:

● None.
Test IKEv2.EN.R.1.1.4.5: Invalid Order Payloads

Purpose:

To verify an IKEv2 device properly handles IKE message with invalid order payloads.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEi payload and Ni payload replace each other.</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)

1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1

The NUT never transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:

- None.
Group 1.5. Cookies

Test IKEv2.EN.R.1.1.5.1: Cookies

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.R.1.1.5.2: Invalid Cookies

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.R.1.1.5.3: Interaction of COOKIE and INVALID KE PAYLOAD

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.R.1.1.5.4: Interaction of COOKIE and INVALID KE PAYLOAD with unoptimized Initiator

This test case was deleted at revision 1.1.0.
Group 1.6. Cryptographic Algorithm Negotiation

Test IKEv2.EN.R.1.1.6.1: Cryptographic Algorithm Negotiation for IKE_SA

Purpose:

To verify an IKEv2 device properly handles various algorithms for IKE_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - From part A to part H, configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>IKE_SA INIT exchanges Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A ENCR_AES_CBC</td>
<td>ENCR_AES_CBC</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part B</td>
<td>DELETED</td>
<td>DELETED</td>
<td>DELETED</td>
<td>DELETED</td>
</tr>
<tr>
<td>Part C ENCR_3DES</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part D ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_AES_XCBC_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part E ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 14</td>
<td></td>
</tr>
<tr>
<td>Part F ENCR_3DES</td>
<td>PRF_HMAC_SHA2_256</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part G ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA2_256_128</td>
<td>Group 24</td>
<td></td>
</tr>
<tr>
<td>Part H ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: IKE_SA_INIT request
Packet #1 is same as Common Packet #1 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform Type ENCR is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>12 (AES CBC)</td>
</tr>
<tr>
<td>SA Attribute</td>
<td>Attribute Type</td>
<td>14 (Key Length)</td>
</tr>
<tr>
<td></td>
<td>Attribute Value</td>
<td>128</td>
</tr>
</tbody>
</table>

Part B:
This test case is deleted at revision 1.0.4.

Part C:
SA Transform of Tranform Type PRF is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>2 (PRF)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>4 (AES128_XCBC)</td>
</tr>
</tbody>
</table>

Part D:
SA Transform of Tranform Type INTEG is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>5 (AES_XCBC 96)</td>
</tr>
</tbody>
</table>

Part E:
SA Transform of Tranform Type D-H is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>4 (D-H)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>14 (2048 MODP Group)</td>
</tr>
</tbody>
</table>

Part F:
SA Transform of Tranform Type PRF is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>2 (PRF)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>5 (HMAC_SHA2 256)</td>
</tr>
</tbody>
</table>

Part G:
SA Transform of Tranform Type INTEG is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
</tbody>
</table>
Part H:
SA Transform of Transform Type D-H is replaced by the following SA Transform.

24. Observe the messages transmitted on Link A.

**Part G: Integrity Algorithm AUTH_HMAC_SHA2_256_128 (ADVANCED)**

25. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
26. Observe the messages transmitted on Link A.
27. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
28. Observe the messages transmitted on Link A.

**Part H: D-H Group Group 24 (ADVANCED)**

29. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
30. Observe the messages transmitted on Link A.
31. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
32. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part B**

This test case is deleted at revision 1.0.4.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_AES128_CBC", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part D**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_AES_XCBC_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part E**

**Step 18: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 14" as accepted algorithms.

**Step 20: Judgment #2**  
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part F**  
**Step 22: Judgment #1**  
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA2_256", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 24: Judgment #2**  
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part G**  
**Step 26: Judgment #1**  
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA2_256_128" and "D-H Group 2" as accepted algorithms.

**Step 28: Judgment #2**  
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part H**  
**Step 30: Judgment #1**  
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 24" as accepted algorithms.

**Step 32: Judgment #2**  
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Possible Problems:**

- None.
Test IKEv2.EN.R.1.1.6.2: Cryptographic Algorithm Negotiation for CHILD_SA

Purpose:

To verify an IKEv2 device properly handles various algorithms for CHILD_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

From part A to part G, TN1 transmits an IKE_AUTH request including a SA payload which contains the transforms as follows:

<table>
<thead>
<tr>
<th>Part</th>
<th>Encryption</th>
<th>Integrity</th>
<th>Extended Sequence Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ENCR AES_CBC</td>
<td>HMAC_SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>B</td>
<td>ENCR AES_CTR</td>
<td>HMAC_SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>C</td>
<td>ENCR NULL</td>
<td>HMAC_SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>D</td>
<td>ENCR 3DES</td>
<td>AES_XCBC 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>E</td>
<td>ENCR 3DES</td>
<td>NONE</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>F</td>
<td>ENCR 3DES</td>
<td>HMAC_SHA1 96</td>
<td>Extended Sequence Numbers</td>
</tr>
<tr>
<td>G</td>
<td>ENCR 3DES</td>
<td>HMAC_SHA2 256 128</td>
<td>No Extended Sequence Numbers</td>
</tr>
</tbody>
</table>

Procedure:
Packet #2: IKE_AUTH request
Packet #2 is same as Common Packet #3 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform Type ENCR is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>SA Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>12 (AES_CBC)</td>
<td>Attribute Type 14 (Key Length) Attribute Value 128</td>
</tr>
</tbody>
</table>

Part B:
SA Transform of Tranform Type ENCR is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>SA Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>13 (AES_CTR)</td>
<td>Attribute Type 14 (Key Length) Attribute Value 128</td>
</tr>
</tbody>
</table>

Part C:
SA Transform of Tranform Type ENCR is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>SA Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>11 (ENCR_NULL)</td>
<td></td>
</tr>
</tbody>
</table>

Part D:
SA Transform of Tranform Type INTEG is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>5 (AES_XCBC_96)</td>
</tr>
</tbody>
</table>

Part E:
SA Transform of Tranform Type INTEG is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>0 (NONE)</td>
</tr>
</tbody>
</table>
Part F:
SA Transform of Transform Type ESN is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>8</td>
<td>5 (ESN)</td>
<td>0</td>
<td>1 (ESN)</td>
</tr>
</tbody>
</table>

Part G:
SA Transform of Transform Type INTEG is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>12 (HMAC_SHA2_256_128)</td>
</tr>
</tbody>
</table>

Part A: Encryption Algorithm ENCR_AES_CBC (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP with the accepted cryptographic suite to the NUT.
6. Observe the messages transmitted on Link A.

Part B: Encryption Algorithm ENCR_AES_CTR (ADVANCED)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP with the accepted cryptographic suite to the NUT.
12. Observe the messages transmitted on Link A.

Part C: Encryption Algorithm ENCR_NULL (ADVANCED)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
16. Observe the messages transmitted on Link A.
17. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP with the accepted cryptographic suite to the NUT.
18. Observe the messages transmitted on Link A.

Part D: Integrity Algorithm AUTH_AES_XCBC_96 (ADVANCED)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP with the accepted cryptographic suite to the NUT.
24. Observe the messages transmitted on Link A.

**Part E: Integrity Algorithm NONE (ADVANCED)**
25. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
28. Observe the messages transmitted on Link A.
29. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP with the accepted cryptographic suite to the NUT.
30. Observe the messages transmitted on Link A.

**Part F: Extended Sequence Numbers (ADVANCED)**
31. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
32. Observe the messages transmitted on Link A.
33. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
34. Observe the messages transmitted on Link A.
35. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP with the accepted cryptographic suite to the NUT.
36. Observe the messages transmitted on Link A.

**Part G: Integrity Algorithm AUTH_HMAC_SHA2_256_128 (ADVANCED)**
37. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
38. Observe the messages transmitted on Link A.
39. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
40. Observe the messages transmitted on Link A.
41. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP with the accepted cryptographic suite to the NUT.
42. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_AES_CBC", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Part B**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 10: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_AES_CTR", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 12: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

---

**Part C**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_NULL", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

---

**Part D**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 22: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 24: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

---

**Part E**

**Step 26: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 28: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "NONE" and "No Extended Sequence Numbers" as accepted algorithms. However, the transform indicating "NONE" can be omitted.

**Step 30: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

---

**Part F**

**Step 32: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 34: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "Extended Sequence Numbers" as accepted algorithms.

**Step 36: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

*Part G*

**Step 38: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 40: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA2_256_128" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 42: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Possible Problems:**

- None.
Test IKEv2.EN.R.1.1.6.3: Receiving Multiple Transforms for IKE_SA

Purpose:

To verify an IKEv2 device properly handles IKE_SA_INIT request with an multiple transforms.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1 See below

From part A to part D, TN1 transmits an IKE_SA_INIT request including a SA payload which contains the transforms as follows:

<table>
<thead>
<tr>
<th>IKE_SA_INIT exchanges Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_AES_CBC, ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES</td>
<td>PRF_AES128_CBC, PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_AES_XCBC_96, AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part D</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 14 or Group 24, Group 2</td>
</tr>
</tbody>
</table>

- Packet #1 IKE_SA_INIT request

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the common packet #1</td>
</tr>
</tbody>
</table>
### Proposal #1

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Proposal Length</th>
<th>Proposal #</th>
<th>Protocol ID</th>
<th>SPI Size</th>
<th># of Transforms</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0</td>
<td>44</td>
<td>1</td>
<td>1 (IKE)</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>According to above configuration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>According to above configuration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
2. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo-Random Functions (BASIC)
3. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
4. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithms (BASIC)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
6. Observe the messages transmitted on Link A.

Part D: Multiple D-H Groups (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload...
8. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Part B**

**Step 4: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Part C**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Part D**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.6.4: Receiving Multiple Proposals for IKE_SA

Purpose:

To verify an IKEv2 device properly handles IKE_SA_INIT request with multiple proposals.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

From part A to part D, TN1 transmits an IKE_SA_INIT request including a SA payload which contains the proposals as follows:

<table>
<thead>
<tr>
<th>IKE_SA_INIT exchanges Algorithms</th>
<th>Proposals</th>
<th>Protocol ID</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>Proposal 1</td>
<td>IKE</td>
<td>ENCR,AES_CBC</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA1,96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part A</td>
<td>Proposal 2</td>
<td>IKE</td>
<td>ENCR,3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA1,96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part B</td>
<td>Proposal 1</td>
<td>IKE</td>
<td>ENCR,3DES</td>
<td>PRF_AES128_CBC</td>
<td>AUTH HMAC_SHA1,96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part B</td>
<td>Proposal 2</td>
<td>IKE</td>
<td>ENCR,3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA1,96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part C</td>
<td>Proposal 1</td>
<td>IKE</td>
<td>ENCR,3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH AES,XCBC,96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part C</td>
<td>Proposal 2</td>
<td>IKE</td>
<td>ENCR,3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA1,96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part D</td>
<td>Proposal 1</td>
<td>IKE</td>
<td>ENCR,3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA1,96</td>
<td>Group 14 or Group 24</td>
</tr>
<tr>
<td>Part D</td>
<td>Proposal 2</td>
<td>IKE</td>
<td>ENCR,3DES</td>
<td>PRF HMAC_SHA1</td>
<td>AUTH HMAC_SHA1,96</td>
<td>Group 2</td>
</tr>
</tbody>
</table>

Packet #1 IKE_SA_INIT request

<p>| IPv6 Header | Same as the Common Packet #1 |
| UDP Header  | Same as the Common Packet #1 |
| IKEv2 Header| Same as the Common Packet #1 |
| SA Payload  | Other fields are same as the common packet #1 |</p>
<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>2 (more)</th>
<th>2 (more)</th>
<th>2 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal Length</td>
<td>44</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protocol ID</td>
<td>1 (IKE)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td># of Transforms</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>1 (ENC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>According to above configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>2 (PRF)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>According to above configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>According to above configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proposal #2</td>
<td>SA Proposal</td>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal Length</td>
<td>44</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protocol ID</td>
<td>1 (IKE)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td># of Transforms</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>1 (ENC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>According to above configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>2 (PRF)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>2 (HMAC,SHA1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>2 (HMAC,SHA1,96)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
2. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo-Random Functions (BASIC)
3. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
4. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithms (BASIC)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
6. Observe the messages transmitted on Link A.

Part D: Multiple D-H Groups (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
8. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Part B
Step 4: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Part C
Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Part D
Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:

• None.
Test IKEv2.EN.R.1.1.6.5: Receiving Multiple Transforms for CHILD_SA

Purpose:

To verify an IKEv2 device properly handles an IKE_AUTH request with multiple transforms.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>IKE_AUTH exchanges</th>
<th>Algorithms</th>
<th>Encryption</th>
<th>Integrity</th>
<th>ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES ENCR_AES_CBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
<td></td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES AUTH_HMAC_SHA1_96 AUTH_AES_XCBC_96</td>
<td>No ESN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES AUTH_HMAC_SHA1_96</td>
<td>No ESN ESN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Packet #1: See Common Packet #1
Packet #2: See below

From part A to part C, TN1 transmits an IKE_AUTH request including a SA payload which contains the transforms as follows:
IPv6 FORUM TECHNICAL DOCUMENT

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Proposal Length</td>
<td>40</td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>4</td>
</tr>
<tr>
<td>SPI</td>
<td>Any</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>5 (ESN)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>5 (ESN)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
4. Observe the messages transmitted on Link A.

Part B: Multiple Integrity Algorithms (BASIC)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
8. Observe the messages transmitted on Link A.
Part C: Multiple Extended Sequence Numbers (BASIC)

9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Part B

Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 8: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Part C

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.6.6: Receiving Multiple Proposals for CHILD_SA

Purpose:

To verify an IKEv2 device properly handles an IKE_AUTH request with multiple proposals.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

IKEv2 exchanges Algorithms

<table>
<thead>
<tr>
<th>Proposal</th>
<th>Protocol</th>
<th>Encryption</th>
<th>Integrity</th>
<th>ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>Proposal #1</td>
<td>ESP</td>
<td>ENCR_AES_CBC</td>
<td>AUTH HMAC SHA1 96</td>
</tr>
<tr>
<td>Part A</td>
<td>Proposal #2</td>
<td>ESP</td>
<td>ENCR_3DES</td>
<td>AUTH HMAC SHA1 96</td>
</tr>
<tr>
<td>Part B</td>
<td>Proposal #1</td>
<td>ESP</td>
<td>ENCR_3DES</td>
<td>AUTH AES XCBC 96</td>
</tr>
<tr>
<td>Part B</td>
<td>Proposal #2</td>
<td>ESP</td>
<td>ENCR_3DES</td>
<td>AUTH HMAC SHA1 96</td>
</tr>
<tr>
<td>Part C</td>
<td>Proposal #1</td>
<td>ESP</td>
<td>ENCR_3DES</td>
<td>AUTH HMAC SHA1 96</td>
</tr>
<tr>
<td>Part C</td>
<td>Proposal #2</td>
<td>ESP</td>
<td>ENCR_3DES</td>
<td>AUTH HMAC SHA1 96</td>
</tr>
</tbody>
</table>

TN1 transmits an IKE_AUTH request including a SA payload which contains the two proposals as follows:

Packet #1 See Common Packet #1
Packet #2 See below
<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
<tr>
<td></td>
<td>SA Proposals: See below</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
</tbody>
</table>

**Proposal #1**

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>2 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Proposal Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td># of Transforms</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>SPI</td>
<td>Any</td>
</tr>
</tbody>
</table>

**SA Transform**

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

**Proposal #2**

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Proposal Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>Proposal #</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td># of Transforms</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>SPI</td>
<td>Any</td>
</tr>
</tbody>
</table>

**SA Transform**

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
</tbody>
</table>
Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
4. Observe the messages transmitted on Link A.

Part B: Multiple Integrity Algorithms (BASIC)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
8. Observe the messages transmitted on Link A.

Part C: Multiple Extended Sequence Numbers (BASIC)
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including a SA Proposal with "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Part B
Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 8: Judgment #2
The NUT transmits an IKE_AUTH response including a SA Proposal with "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Part C
Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Transform Length 8
Transform Type 5 (ESN)
Reserved 0
Transform ID 0 (No ESN)
Step 12: Judgment #2
The NUT transmits an IKE_AUTH response including a SA Proposal with "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.6.7: Sending INVALID_KE_PAYLOAD

Purpose:

To verify an IKEv2 device properly handles a KE payload which has different D-H Group # from accepted D-H Group #.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. Enable PFS.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>See Common Packet #1</td>
</tr>
<tr>
<td>#2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>#3</td>
<td>See below</td>
</tr>
<tr>
<td>#4</td>
<td>See below</td>
</tr>
</tbody>
</table>
Packet #3: CREATE_CHILD_SA request for rekeying CHILD_SA

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #13</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See SA Table below</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>Next Payload</td>
<td>34 (KE)</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>Next Payload 44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>264</td>
</tr>
<tr>
<td>DH Group #</td>
<td>14</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Key Exchange Data</td>
<td>DH#14 public key value</td>
</tr>
<tr>
<td>TSI Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
</tbody>
</table>

SA Payloads

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Proposal Length</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1 (IKE)</td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td># of Transforms</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>2 (PRF)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC SHA1)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC SHA1 96)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>4 (D-H)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (1024 MODP Group)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>4 (D-H)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>14 (2048 MODP Group)</td>
<td></td>
</tr>
</tbody>
</table>

Packet #4: CREATE_CHILD_SA request for rekeying CHILD_SA

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Other fields are same as the Common Packet #13</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as Packet #3</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>Next Payload</td>
<td>34 (KE)</td>
</tr>
<tr>
<td>Kei Payload</td>
<td>Other fields are same as the Packet #3</td>
</tr>
<tr>
<td>DH Group #</td>
<td>2</td>
</tr>
<tr>
<td>Key Exchange Data</td>
<td>DH#2 public key value</td>
</tr>
<tr>
<td>Tsi Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>Tsr Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs. The CREATE_CHILD_SA contains a D-H Group transform to use D-H Group 2 and D-H Group 14, and a Key Exchange payload which contains 14 (D-H Group 14) as DH Group # field and the Key Exchange Data. It is possible to use D-H Group 24 instead of D-H Group 14.
6. Observe the messages transmitted on Link A.
7. After reception of CREATE_CHILD_SA response indicating INVALID_KE_PAYLOAD from the NUT, TN1 retransmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs. The CREATE_CHILD_SA request contains a D-H Group transform to use D-H Group 2 and D-H Group 14, and a Key Exchange payload which contains 2 (D-H Group 2) as DH Group # field and the Key Exchange Data. It is possible to use D-H Group 24 instead of D-H Group 14.
8. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including a Notify payload of type INVALID_KE_PAYLOAD which contains 2 (D-H Group 2) as Notification Data.

**Step 8: Judgment #4**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers" and "D-H Group 2" as proposed algorithms.

**Possible Problems:**
• None.
Test IKEv2.EN.R.1.1.6.8: Sending INVALID_KE_PAYLOAD in Initial Exchange

Purpose:

To verify an IKEv2 device properly handles KE payload which has different D-H Group # from accepted D-H Group #.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See below</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common packet #1</td>
</tr>
</tbody>
</table>

Packet #1: IKE_SA_INIT request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See SA Table below</td>
</tr>
<tr>
<td>DH Group #</td>
<td>14</td>
</tr>
<tr>
<td>Key Exchange Data</td>
<td>DH#14 public key value</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
</tbody>
</table>

SA Payloads
SA Proposal
Next Payload 0 (last)
Reserved 0
Proposal Length 48
Proposal # 1
Protocol ID 1 (IKE)
SPI Size 0
# of Transforms 5
SA Transform
Next Payload 3 (more)
Reserved 0
Transform Length 8
Transform Type 1 (ENCR)
Reserved 0
Transform ID 3 (3DES)
SA Transform
Next Payload 3 (more)
Reserved 0
Transform Length 8
Transform Type 2 (PRF)
Reserved 0
Transform ID 2 (HMAC_SHA1)
SA Transform
Next Payload 3 (more)
Reserved 0
Transform Length 8
Transform Type 3 (INTEG)
Reserved 0
Transform ID 2 (HMAC_SHA1_96)
SA Transform
Next Payload 3 (more)
Reserved 0
Transform Length 8
Transform Type 4 (D-H)
Reserved 0
Transform ID 2 (1024 MODP Group)
SA Transform
Next Payload 0 (last)
Reserved 0
Transform Length 8
Transform Type 4 (D-H)
Reserved 0
Transform ID 14 (2048 MODP Group)

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload which contains a D-H Group transform proposes using D-H Group 2 and D-H Group 14, and a Key Exchange payload which contains 14 (D-H Group 14) as DH Group # field and the Key Exchange Data. It is possible to use D-H Group 24 instead of D-H Group 14.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT request including KE payload with D-H Group 2 public key value to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including a Notify payload of type INVALID_KE_PAYLOAD which contains 2 (D-H Group 2) as Notification Data. The message’s IKE_SA Responder’s SPI value is set to zero.

Step 4: Judgment #2
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.
Possible Problems:

- None.
Test IKEv2.EN.R.1.1.6.9: Creating an IKE_SA without a CHILD_SA

Purpose:
To verify that an IKEv2 device can handle a failure of creating a CHILD_SA during the IKE_AUTH exchange.

References:
- [RFC 4718] - Sections 4.2

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH request
Packet #2 is same as Common Packet #3 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform Type ENCR is replaced by the following SA Transform.
<table>
<thead>
<tr>
<th>Reserved</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>12 (AES_CBC)</td>
</tr>
<tr>
<td>SA Attribute</td>
<td>Attribute Type</td>
</tr>
<tr>
<td>Attribute Value</td>
<td>128</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request with unacceptable SA proposal for the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an INFORMATIONAL request with no payloads.
6. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including a Notify type of NO_PROPOSAL_CHOSEN.

**Step 6: Judgment #3**
The NUT transmits an INFORMATIONAL response followed by an Encrypted payload with no payloads contained in it.

**Possible Problems:**

- None
Group 1.7. Traffic Selector Negotiation

Test IKEv2.EN.R.1.1.7.1: Narrowing Traffic Selectors

Purpose:

To verify an IKEv2 device allows the responder to choose a subset of the traffic proposed by the initiator.

References:

- [RFC4306] - Section 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration except Traffic Selector. Traffic Selector should be configured as following.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Address Range</td>
<td>Next Layer Protocol</td>
</tr>
<tr>
<td>Inbound</td>
<td>TN1</td>
<td>TCP</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT</td>
<td>TCP</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
<tr>
<td>Tsr Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
</tbody>
</table>

**Traffic Selectors**

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tsr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
</tbody>
</table>
Part A (BASIC)

1. TN1 sends an IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 sends an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a TCP-SYN packet with IPsec ESP using corresponding algorithms to closed port on NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
8. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. The Traffic Selector is narrowed to allow only TCP (6) as IP Protocol.

Step 6: Judgment #3
The NUT transmits a TCP-RST packet with IPsec ESP using corresponding algorithms.

Step 8: Judgment #4
The NUT never transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.7.2: TS_UNACCEPTABLE

Purpose:

To verify an IKEv2 device properly handles the Traffic Selector.

References:

- [RFC 4306] - Sections 2.8 and 3.10.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration except Traffic Selector. Traffic Selector should be configured as following.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Address Range</td>
<td>Next Layer Protocol</td>
</tr>
<tr>
<td>Inbound</td>
<td>TN1</td>
<td>TCP</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT</td>
<td>TCP</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
<tr>
<td>Field</td>
<td>Content</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

- Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Field</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #7</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>58 (ICMPv6)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS Type</td>
<td>8 (IPV6_ADDR_RANGE)</td>
</tr>
<tr>
<td>IP Protocol ID</td>
<td>58 (ICMPv6)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>
**Part A (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request including ICMPv6 (58) as IP Protocol ID value in Traffic Selector Payload to create new CHILD_SA.
6. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including a Notify payload of type TS_UNACCEPTABLE.

**Possible Problems:**

- None.
Test IKEv2.EN.R.1.1.7.3: Narrowing Traffic Selectors from multiple Traffic Selector

Purpose:

To verify an IKEv2 device allows the responder to choose a subset of the traffic proposed by the initiator.

References:

- [RFC4306] - Section 2.8
- [RFC4718] - Section 4.10

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration except Traffic Selector. Traffic Selector should be configured as following.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Address Range</td>
<td>Next Layer Protocol</td>
</tr>
<tr>
<td>Inbound</td>
<td>TN1</td>
<td>TCP</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT</td>
<td>TCP</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IDI Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Traffic Selectors: See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Traffic Selectors: See below</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>58 (IPV6-ICMP)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TN1’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>
Packet #3: TCP SYN packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
<th>NUT’s Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
<td>CHILD_SA’s SPI value used by this message</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Next Header</td>
<td>6 (TCP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Integrity Check Value</td>
<td>The cryptographic checksum of the entire message</td>
<td></td>
</tr>
<tr>
<td>TCP Header</td>
<td>Source Port</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Flags</td>
<td>SYN (0x02)</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. TN1 sends an IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 sends an IKE_AUTH request to the NUT. The message includes two Traffic Selectors. One is set to 6 (TCP) as IP Protocol. Another is set to 58 (IPV6-ICMP).
4. Observe the messages transmitted on Link A.
5. TN1 transmits a TCP-SYN packet with IPsec ESP using corresponding algorithms to close port on NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
8. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. The Traffic Selector Payload has one Traffic Selector with IP Protocol 6 (TCP) to narrow the proposed Traffic Selectors.
Step 6: Judgment #3
The NUT transmits a TCP-RST packet with IPsec ESP using corresponding algorithms.

Step 8: Judgment #4
The NUT never transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

• None.
Group 1.8. Error Handling

Test IKEv2.EN.R.1.1.8.1: INVALID_IKE_SPI

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.R.1.1.8.2: INVALID SYNTAX

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.R.1.1.8.3: INVALID_SELECTORS

This test case was deleted at revision 1.1.0.
Group 1.10. Authentication of the IKE_SA

Test IKEv2.EN.R.1.1.10.1: Sending Certificate Payload

Purpose:

To verify an IKEv2 device handles a CERTREQ payload and transmits a CERT payload properly.

References:

- [RFC 4306] - Sections 1.2 and 3.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Part</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>NUT’s global address on Link A</td>
</tr>
<tr>
<td>B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>nut.example.com</td>
</tr>
<tr>
<td>C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:nut@example.com">nut@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

- Packet #1: See Common Packet #1
- Packet #2: See below

- Packet #2: IKE_AUTH request
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td>CERTREQ Payload</td>
<td>See below</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>CERTREQ Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>Any</td>
</tr>
<tr>
<td>Certificate Encoding</td>
<td>4 (X.509 Certificate - Signature)</td>
</tr>
<tr>
<td>Certificate Authority</td>
<td>Any</td>
</tr>
</tbody>
</table>

**Part A: ID_IPV6_ADDR (ADVANCED)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with a CERTREQ payload to the NUT.
4. Observe the messages transmitted on Link A.

**Part B: ID_FQDN (ADVANCED)**
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with a CERTREQ payload to the NUT.
8. Observe the messages transmitted on Link A.

**Part A: ID_RFC822_ADDR (ADVANCED)**
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with a CERTREQ payload to the NUT.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response. The response includes an ID payload with ID_IPV6_ADDR and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH response. The response includes an ID payload with ID_FQDN and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Part B**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH response. The response includes an ID payload with ID_RFC822_ADDR and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Possible Problems:**

- None.
Test IKEv2.EN.R.1.1.10.2: Sending Certificate Request Payload

Purpose:

To verify an IKEv2 device properly transmits CERTREQ payload.

References:

- [RFC 4306] - Sections 1.2 and 3.7

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.

- Configuration
  - In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Remote</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>TN1’s global address on Link A</td>
</tr>
<tr>
<td>Part B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>tn.example.com</td>
</tr>
<tr>
<td>Part C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:tn@example.com">tn@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1  See Common Packet #1

Part A: ID_IPV6_ADDR (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Part B: ID_FQDN (ADVANCED)
3. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
4. Observe the messages transmitted on Link A.

Part C: ID_RFC822_ADDR (ADVANCED)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Part B

Step 4: Judgment #1
The NUT transmits an IKE_SA_INIT response with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Part C

Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.10.3: RSA Digital Signature

Purpose:

To verify an IKEv2 device authenticates the corresponding node by RSA Digital Signature.

References:

- [RFC 4306] - Sections 1.2 and 3.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Part</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>TN1’s global address on Link A</td>
</tr>
<tr>
<td>B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>tn.example.com</td>
</tr>
<tr>
<td>C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:tn@example.com">tn@example.com</a></td>
</tr>
</tbody>
</table>

Pre-Sequence and Cleanup Sequence
IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>Packet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #1</td>
<td>See Common Packet #1</td>
</tr>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Header
Same as the Common Packet #3

UDP Header
Same as the Common Packet #3

IKEv2 Header
Same as the Common Packet #3

E Payload
Same as the Common Packet #3

IDi Payload
Next Payload 37 (CERT)
Other fields are same as the Common Packet #3

CERT Payload
See below

AUTH Payload
Same as the Common Packet #3

N Payload
Same as the Common Packet #3

SA Payload
Same as the Common Packet #3

TSi Payload
Same as the Common Packet #3

TSr Payload
Same as the Common Packet #3

Next Payload 39 (AUTH)
Critical 0
Reserved 0
Payload Length Any
Certificate Encoding 4 (X.509 Certificate – Signature)
Certificate Data any

Part A: ID_IPV6_ADDR (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with an IDi payload as described above and a CERT payload to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.

Part B: ID_FQDN (ADVANCED)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with an IDi payload as described above and a CERT payload to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
12. Observe the messages transmitted on Link A.

Part C: ID_RFC822_ADDR (ADVANCED)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with an IDi payload as described above and a CERT payload to the NUT.
16. Observe the messages transmitted on Link A.
17. After reception of IKE_AUTH response from the NUT, TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
18. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Part B**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 10: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 12: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Part C**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
IPv6 FORUM TECHNICAL DOCUMENT

Test IKEv2.EN.R.1.1.10.4: HEX string PSK

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key.

References:

- [RFC 4306] - Sections 2.15

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Authentication Key Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local 0xabdcafeabadcafeabadcafeabadcafe (128 bit binary string)</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:
Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Possible Problems:

- None.
Group 1.11 Invalid Values

Test IKEv2.EN.R.1.1.11.1: Non zero RESERVED fields in IKE_SA_INIT request

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>All RESERVED fields are set to one.</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.11.2: Non zero RESERVED fields in IKE_AUTH request

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.</td>
</tr>
<tr>
<td>4. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Possible Problems:**

- None.
Test IKEv2.EN.R.1.1.11.3: Version bit is set

Purpose:
To verify an IKEv2 device ignores the content of Version bit in IKE messages.

References:
- [RFC 4306] - Sections 3.1

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A (BASIC)
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request whose Version bit is set to one.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:
- None.
Test IKEv2.EN.R.1.1.11.4: Response bit is set

Purpose:

To verify an IKEv2 device ignores an IKE request message whose Response bit is set.

References:

- [RFC 4306] - Sections 2.21

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response bit is set to one.</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request whose Response bit is set to one.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT never responds with an IKE_SA_INIT response to an IKE_SA_INIT request from the TN1.

Possible Problems:

- None.
Test IKEv2.EN.R.1.1.11.5: Unrecognized Notify Message Type

Purpose:

To verify an IKEv2 device ignores the unrecognized Notify Message Type in IKE messages.

References:

- [RFC 4306] - Sections 3.10.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>All fields are same as Common Packet #3</td>
<td></td>
</tr>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #3</td>
<td></td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #3</td>
<td></td>
</tr>
<tr>
<td>E Payload</td>
<td>All fields are same as Common Packet #3</td>
<td></td>
</tr>
<tr>
<td>Idi Payload</td>
<td>All fields are same as Common Packet #3</td>
<td></td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>All fields are same as Common Packet #3</td>
<td></td>
</tr>
<tr>
<td>N Payload</td>
<td>All fields are same as Common Packet #3</td>
<td></td>
</tr>
</tbody>
</table>
Part A: Unrecognized Notify Message Type of error 16383 (BASIC)
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request with a Notify payload of unrecognized Notify Message Type value (16383) to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.

Part B: Unrecognized Notify Message Type of status 65535 (BASIC)
7. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request with a Notify payload of unrecognized Notify Message Type value (65535) to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Part B

Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.
Step 10: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 12: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Group 2. The CREATE_CHILD_SA Exchange

Group 2.1. Header and Payload Formats

Test IKEv2.EN.R.1.2.1.1: Receipt of CREATE_CHILD_SA request

Purpose:

To verify an IKEv2 device transmits a CREATE_CHILD_SA response using properly Header and Payloads format

References:

- [RFC 4306] - Sections 1.3 and 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #13</td>
</tr>
</tbody>
</table>
Part A: IKE Header Format (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
6. Observe the messages transmitted on Link A.

Part B: Encrypted Payload Format (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
12. Observe the messages transmitted on Link A.

Part D: Notify Payload (USE_TRANSPORT_MODE) Format (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
18. Observe the messages transmitted on Link A.

Part E: SA Payload Format (BASIC)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
24. Observe the messages transmitted on Link A.

Part F: Nonce Payload Format (BASIC)
25. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
28. Observe the messages transmitted on Link A.
29. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
30. Observe the messages transmitted on Link A.

Part G: TSi Payload Format (BASIC)
31. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
32. Observe the messages transmitted on Link A.
33. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
34. Observe the messages transmitted on Link A.
35. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
36. Observe the messages transmitted on Link A.

Part H: TSr Payload Format (BASIC)
37. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
38. Observe the messages transmitted on Link A.
39. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
40. Observe the messages transmitted on Link A.
41. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
42. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including properly formatted IKE Header containing following values:

- An IKE_SA Initiator’s SPI field is set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field is set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field is set to Encrypted Payload (46).
• A Major Version field is set to 2.
• A Minor Version field is set to zero.
• An Exchange Type field is set to CREATE_CHILD_SA (36).
• A Flags field is set to (00000100)2 = (4)10.
• A Message ID field is set to the same value as corresponding IKEv2 request message’s Message ID.
• A Length field is set to the length of the message (header + payloads) in octets.

Part B

Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 10: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 12: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including properly formatted Encrypted Payload containing following values:

A Next Payload field is set to N Payload (41).
• A Critical field is set to zero.
• A RESERVED field is set to zero.
• A Payload Length field is set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Check sum Data.
• An Initialization Vector field is set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
• An Encrypted IKE Payloads field is set to subsequent payloads encrypted by ENCR_3DES.
• A Padding field is set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
• A Pad Length field is set to the length of the Padding field.
• An Integrity Checksum Data set to the cryptographic checksum of the entire payload.
message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum
must be valid by calculation according to the manner described in RFC.

Part C

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted
algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 18: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including properly formatted Notify
Payload containing following values:

- A Next Payload field is set to SA Payload (33).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload. It is 8 bytes for
  USE_TRANSPORT_MODE.
- A Protocol ID field is set to undefined (0).
- A SPI Size field is set to zero.
- A Notify Message Type field is set to USE_TRANSPORT_MODE (16391)

Part D

Step 20: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted
algorithms.

Step 22: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 24: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field is set to Nr Payload (40).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.

The following proposal must be included in Proposals field.
Proposal #1
- A 0 or 2 field is set to zero if this structure is the last proposal, otherwise set to 2.
- A RESERVD field is set to zero.
- A Proposal Length field is set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field is set to 1.
- A Protocol ID field is set to ESP (3).
- A SPI Size field is set to 4.
- A # of Transforms field is set to 3.
- A SPI field is set to the sending entity’s SPI (4 octets value)

Transform field is set to following (There are 3 Transform Structures).

Transform #1
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field is set to ENCR (1).
- A RESERVED field is set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field is set to INTEG (3).
- A RESERVED field is set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field is set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field is set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
- A Transform Type field is set to ESN (5).
- A RESERVED field is set to zero.
- A Transform ID set to No Extended Sequence Numbers (0).

Part E

Step 26: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 28: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 30: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including properly formatted Nonce Payload containing following values:

- A Next Payload field is set to TSi Payload (44).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Nonce Data field is set to random data generated by the transmitting entity.
- The size of the Nonce must between 16 and 256 octets.

Figure 77 Nonce Payload format

Part F

Step 32: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 34: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 36: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including properly formatted TSi Payload containing following values:

![Figure 78 TSi Payload format](image)

- A Next Payload field is set to TSr Payload (45).
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Number of TSs field is set to 1.
- A RESERVED field is set to zero.

The following traffic selector must be included in Traffic Selectors field.

![Figure 79 Traffic Selector](image)

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header.
• A Start Port field is set to zero.
• An End Port field is set to 65535.
• A Starting Address field is set to TN1 address.
• A Ending Address field is set to TN1 address.

**Part G**

**Step 38: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 40: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 42: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including properly formatted TSr Payload containing following values:

![Figure 80 TSr Payload format](image)

- A Next Payload field is set to zero.
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length of the current payload.
- A Number of TSs field is set to 1.
- A RESERVED field is set to zero.

The following traffic selector must be included in Traffic Selectors field.
Figure 81 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field is set to zero.
- A Selector Length field is set to length of this Traffic Selector Substructure including the header.
- A Start Port field is set to zero.
- An End Port field is set to 65535.
- A Starting Address field is set to NUT address.
- An Ending Address field is set to NUT address.

Possible Problems:

- CREATE_CHILD_SA response has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- Each of transforms can be located in the any order.
Group 2.2. Use of Retransmission Timers

Test IKEv2.EN.R.1.2.2.1: Receipt of retransmitted CREATE_CHILD_SA request

Purpose:
To verify an IKEv2 device retransmits CREATE_CHILD_SA request using properly Header and Payloads format

References:
- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Connect the devices according to the Common Topology.</td>
</tr>
<tr>
<td>2</td>
<td>In each part, configure the devices according to the Common Configuration.</td>
</tr>
<tr>
<td>3</td>
<td>IKEv2 on the NUT is disabled after each part.</td>
</tr>
</tbody>
</table>

...
Packet #1 | See Common Packet #1
---|---
Packet #2 | See Common Packet #3
Packet #3 | See Common Packet #13
Packet #4 | See Common Packet #13

**Part A: (BASIC)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request to rekey the established CHILD_SAs to the NUT.
6. Observe the messages transmitted on Link A.
7. Observe the messages transmitted on Link A.
8. TN1 retransmits the same message as a CREATE_CHILD_SA request transmitted in Step 5 to the NUT.
9. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #4**
The NUT never retransmits a CREATE_CHILD_SA response which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

**Step 9: Judgment #5**
The NUT retransmits a CREATE_CHILD_SA response which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

**Possible Problems:**

• none
Group 2.3. State Synchronization and Connection Timeouts

Test IKEv2.EN.R.1.2.3.1: Receiving Delete Payload for Multiple CHILD_SA

Purpose:

To verify an IKEv2 device transmits a Delete Payload, when CHILD_SAs are deleted.

References:

- [RFC 4306] - Sections 2.4 and 3.11

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common below</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common below</td>
</tr>
</tbody>
</table>
### Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Traffic Selectors See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #3</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Traffic Selectors See below</td>
</tr>
</tbody>
</table>

#### TSi Payload Traffic Selector
- **TS Type**: IPv6_ADDR_RANGE (8)
- **IP Protocol ID**: 6 (TCP)
- **Selector Length**: 40
- **Start Port**: 0
- **End Port**: 65535
- **Starting Address**: TN1’s Global Address on Link X
- **Ending Address**: TN1’s Global Address on Link X

#### TSr Payload Traffic Selector
- **TS Type**: IPv6_ADDR_RANGE (8)
- **IP Protocol ID**: 6 (TCP)
- **Selector Length**: 40
- **Start Port**: 0
- **End Port**: 65535
- **Starting Address**: NUT’s Global Address on Link A
- **Ending Address**: NUT’s Global Address on Link A

### Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #7</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #7</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Traffic Selectors See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #7</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Traffic Selectors See below</td>
</tr>
</tbody>
</table>

#### TSi Payload Traffic Selector
- **TS Type**: IPv6_ADDR_RANGE (8)
- **IP Protocol ID**: 58 (ICMPv6)
- **Selector Length**: 40
- **Start Port**: 0
- **End Port**: 65535
- **Starting Address**: TN1’s Global Address on Link X
- **Ending Address**: TN1’s Global Address on Link X

#### TSr Payload Traffic Selector
- **TS Type**: IPv6_ADDR_RANGE (8)
- **IP Protocol ID**: 58 (ICMPv6)
- **Selector Length**: 40
- **Start Port**: 0
- **End Port**: 65535
- **Starting Address**: NUT’s Global Address on Link A
IPv6 FORUM TECHNICAL DOCUMENT

Packet #4: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
</tr>
<tr>
<td></td>
<td># of SPIs</td>
</tr>
<tr>
<td></td>
<td>Security Parameter Index(es) (SPI)</td>
</tr>
<tr>
<td></td>
<td>SPI negotiated by Initial Exchange</td>
</tr>
<tr>
<td></td>
<td>SPI negotiated by CREATE_CHILD_SA exchange</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request to establish a new CHILD_SA to the NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits an INFORMATIONAL request with a Delete payload including the first negotiated CHILD_SA’s inbound SPI and the second negotiated CHILD_SA’s inbound SPI.
8. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 8: Judgment #4**
The NUT transmits an INFORMATIONAL response with delete payload for SPIs which are negotiated by Initial Exchange and CREATE_CHILD_SA exchange.

**Possible Problems:**

- INFORMATIONAL response from NUT may not contain Delete Payload by implementation policy. This behavior is defined at section 1.4 in RFC 4306 as an
exception.
Group 2.4. Cryptographic Algorithm Negotiation

Test IKEv2.EN.R.1.2.4.1: Sending NO_PROPOSAL_CHOSEN

Purpose:
To verify an IKEv2 device properly handles CREATE_CHILD_SA request with an unacceptable SA payload.

References:
- [RFC 4306] - Sections 2.7 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>
Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Packet Fields</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See below</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>Tsr Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Proposal Length</th>
<th>36</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
<td>1</td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
<td>4</td>
<td># of Transforms</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI</td>
<td>any</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform ID</td>
<td>12 (AES_CBC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform ID</td>
<td>5 (AES_XCBC_96)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Type</td>
<td>5 (ESN)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform ID</td>
<td>1 (ESN)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request to rekey the established CHILD_SAs to the NUT. The CREATE_CHILD_SA request includes a SA payload with a proposal unacceptable by the NUT.
6. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT does not transmit a CREATE_CHILD_SA response or transmits a CREATE_CHILD_SA response including a Notify payload of type NO_PROPOSAL_CHOSEN.

**Possible Problems:**

- None.
Group 2.5. Rekeying CHILD_SA Using a CREATE_CHILD_SA exchange

Test IKEv2.EN.R.1.2.5.1: Close the replaced CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the CREATE_CHILD_SA Exchanges to rekey CHILD_SA and INFORMATIONAL Exchanges to delete old CHILD_SAs.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: See Common Packet #1
Packet #2: See Common Packet #3
Packet #3: See Common Packet #19
Packet #4: See Common Packet #13
Packet #5: See below

Packet #5: INFORMATIONAL request

IPv6 Header | Same as the Common Packet #17
UDP Header | Same as the Common Packet #17
IKEv2 Header | Same as the Common Packet #17
E Payload | Other fields are same as the Common Packet #17
Delete Payload | 
Next Payload | 42 (Delete)
Critical | 0
Reserved | 0
Payload Length | 12
Protocol ID | 3 (ESP)
SPI Size | 4
# of SPIs | 1
Security Parameter Index(es) (SPI) | SPI negotiated by Initial Exchange

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA's SPI value in the SPI field to the NUT.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request including a Delete payload with the old CHILD_SA's SPI value to the NUT.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 8: Judgment #4
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 10: Judgment #5**

The NUT transmits an INFORMATIONAL response including a Delete payload with the old CHILD_SA’s SPI value to the TN1.

**Possible Problems:**

- none
Test IKEv2.EN.R.1.2.5.2: Use of the new CHILD_SA

Purpose:
To verify an IKEv2 device properly handle old CHILD_SA and new CHILD_SA.

References:
- [RFC 4306] - Sections 2.8

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: See Common Packet #1
Packet #2: See Common Packet #3
Packet #3: See Common Packet #19
(CHILD_SA is negotiated by steps 1 through 4.)
Packet #4: See Common Packet #13
Packet #5: See below
Packet #6: See Common Packet #19
(CHILD_SA is negotiated by steps 7 through 8.)

● Packet #5: INFORMATIONAL request

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td>Delete Payload</td>
<td>Next Payload 42 (Delete)</td>
</tr>
<tr>
<td></td>
<td>Next Payload 0 (last)</td>
</tr>
<tr>
<td></td>
<td>Critical 0</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Payload Length 12</td>
</tr>
<tr>
<td></td>
<td>Protocol ID 3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size 4</td>
</tr>
<tr>
<td></td>
<td># of SPI's 1</td>
</tr>
<tr>
<td>Security Parameter Index(es) (SPI)</td>
<td>SPI negotiated by Initial Exchange</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request including a Delete payload with the old CHILD_SA’s SPI value to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to the NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.
**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 8: Judgment #4**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 10: Judgment #5**
The NUT transmits an INFORMATIONAL response including a Delete payload with the old CHILD_SA’s SPI value to the TN1.

**Step 12: Judgment #6**
The NUT transmits an Echo Reply with IPsec ESP using the newly negotiated algorithms.

**Possible Problems:**

- none
Test IKEv2.EN.R.1.2.5.3: Receiving Multiple Transform

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with multiple transforms to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.7, 2.8 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

From part A to part C, TN1 transmits a CREATE_CHILD_SA request including a SA payload which contains the transforms as follows:

<table>
<thead>
<tr>
<th>Part A</th>
<th>CREATE_CHILD_SA exchanges Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Encryption</td>
</tr>
<tr>
<td></td>
<td>ENCR_3DES</td>
</tr>
<tr>
<td></td>
<td>ENCR_AES_CBC</td>
</tr>
</tbody>
</table>
Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Proposal Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Proposal #</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td># of Transforms</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SPI</td>
<td>Any</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>According to above configuration</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>5 (ESN)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

Part B: Multiple Integrity Algorithms (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

Part C: Multiple Extended Sequence Numbers (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part B

Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 10: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 12: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Part C

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 18: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

• none
Test IKEv2.EN.R.1.2.5.4: Receiving Multiple Proposal

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with multiple transforms to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.7, 2.8 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>CREATE_CHILD_SA exchanges Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal</td>
</tr>
<tr>
<td>Part A</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

TN1 transmits a CREATE_CHILD_SA request including a SA payload which contains the two proposals as follows:

Packet #1  See Common Packet #1
Packet #2  See Common Packet #3
Packet #3  See below
Proposal #1: ESP ENCR_3DES AUTH_AES_XCBC_96 No ESN
Proposal #2: ESP ENCR_3DES AUTH HMAC_SHA1_96 No ESN
Proposal #1: ESP ENCR_3DES AUTH HMAC SHA1_96 ESN
Proposal #2: ESP ENCR_3DES AUTH HMAC_SHA1_96 No ESN

• Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #13</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #13</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See below</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
</tbody>
</table>

Proposal #1 SA Proposal

<table>
<thead>
<tr>
<th>Proposal Length</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>4</td>
</tr>
</tbody>
</table>

SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>2 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

Proposal #2 SA Proposal

<table>
<thead>
<tr>
<th>Proposal Length</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal ID</td>
<td>2 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>4</td>
</tr>
</tbody>
</table>

SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>
Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

Part B: Multiple Integrity Algorithms (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

Part C: Multiple Extended Sequence Numbers (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3

The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Part B**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 10: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 12: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Part C**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- none
Test IKEv2.EN.R.1.2.5.5: Perfect Forward Secrecy

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA exchange when Perfect Forward Secrecy enables.

References:

- [RFC 4306] - Sections 2.12

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration. Enable PFS.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
### Packet #4: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th></th>
<th>Packet #1</th>
<th>Packet #2</th>
<th>Packet #3</th>
<th>Packet #4</th>
<th>Packet #5</th>
<th>Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>See Common Packet #1</td>
<td>See Common Packet #3</td>
<td>See Common Packet #19</td>
<td>(CHILD_SA is negotiated by steps 1 through 4.)</td>
<td>See below</td>
<td>See below</td>
</tr>
<tr>
<td>(CHILD_SA is negotiated by steps 7 through 8.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Packet #4: CREATE_CHILD_SA response**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>F Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #13</td>
</tr>
<tr>
<td>Ni Payload</td>
<td>Next Payload 34 (KE)</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>Next Payload 44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>136</td>
</tr>
<tr>
<td>DH Group #</td>
<td>2</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
</tbody>
</table>
Packet #5: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td>Next Payload</td>
<td>42 (Delete)</td>
</tr>
<tr>
<td>Delete Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>12</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of SPIs</td>
<td>1</td>
</tr>
<tr>
<td>Security Parameter Index(es) (SPI)</td>
<td>SPI negotiated by Initial Exchange</td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request including a Delete payload with the old CHILD_SA’s SPI value to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an Echo Request with IPsec ESP using the second negotiated algorithms to the NUT.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 8: Judgment #4**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Step 10: Judgment #5
The NUT transmits an INFORMATIONAL response including a Delete payload with the old CHILD_SA’s SPI value to the TN1.

Step 12: Judgment #6
The NUT transmits an Echo Reply with IPsec ESP using the newly negotiated algorithms.

Possible Problems:

- none
Test IKEv2.EN.R.1.2.5.6: Use of the old CHILD_SA

Purpose:

To verify an IKEv2 device properly handle old CHILD_SA and new CHILD_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

(CHILD_SA is negotiated by steps 1 through 4.)
Packet #4  See Common Packet #13
Packet #5  See Common Packet #19
(CHILD_SA is negotiated by steps 1 through 4.)

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms again.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 8: Judgment #4
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 10: Judgment #5
The NUT transmits an Echo Reply with IPsec ESP. The NUT can use both the first CHILD_SA and the new CHILD_SA.

Possible Problems:

- none
Group 2.6. Rekeying IKE_SAs Using a CREATE_CHILD_SA exchange

Test IKEv2.EN.R.1.2.6.1: Sending CREATE_CHILD_SA response

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA Exchange to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8 and 2.18

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>
**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request including a SA payload. A proposal in the SA payload contains 1 (IKE) in the Protocol ID field, 8 in the SPI size field and the rekeyed IKE_SA’s initiator’s SPI value.
8. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 8: Judgment #4**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s responder’s SPI value in the SPI field.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Test IKEv2.EN.R.1.2.6.2: Receipt of cryptographically valid message on the old SA

Purpose:

To verify an IKEv2 device properly uses old IKE-SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request to the NUT.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request with no payloads protected by the old IKE_SA.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 8: Judgment #4
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s responder’s SPI value in the SPI field.

Step 10: Judgment #5
The NUT responds with an INFORMATIONAL response with no payloads protected by the old IKE_SA.

Possible Problems:

• none
Test IKEv2.EN.R.1.2.6.3: Receipt of cryptographically valid message on the new SA

Purpose:

To verify an IKEv2 device properly uses new IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>Packet #2</th>
<th>Packet #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>See Common Packet #1</td>
<td>See Common Packet #3</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>
Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request to the NUT.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request with no payloads protected by the new IKE_SA and the Message ID field in the IKE header is zero.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 8: Judgment #4
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s responder’s SPI value in the SPI field.

Step 10: Judgment #5
The NUT responds with an INFORMATIONAL response with no payloads protected by the new IKE_SA and the Message ID field in the IKE header is zero.

Possible Problems:

- none
Test IKEv2.EN.R.1.2.6.4: Close the replaced IKE_SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8
- [RFC 4718] - Sections 5.8 and 5.11

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1 See Common Packet #1
Packet #2 See Common Packet #3
Packet #3 See Common Packet #19
Packet #4 See Common Packet #11
Packet #5 See below
Packet #6 See Common Packet #19

- Packet #5: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td>Delete Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
</tr>
<tr>
<td></td>
<td># of SPIs</td>
</tr>
<tr>
<td></td>
<td>Security Parameter Index(es) (SPI)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request to rekey IKE_SA to the NUT.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request with a Delete payload which has 1 (IKE_SA) in the Protocol ID field, zero in the SPI Size field and zero in the # of SPIs field.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms inherited from the replaced IKE_SA.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.
Step 8: Judgment #4
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s responder’s SPI value in the SPI field.

Step 10: Judgment #5
The NUT responds with an INFORMATIONAL response with no payloads.

Step 12: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms inherited from the replaced IKE_SA.

Possible Problems:

- none.
Test IKEv2.EN.R.1.2.6.5: Receiving Multiple Transform

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with multiple transform to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.7, 2.8 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

From part A to part D, TN1 transmits an IKE_SA_INIT request including a SA payload which contains the transforms as follows:

<table>
<thead>
<tr>
<th>Part</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ENCR_AES_CBC</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>B</td>
<td>ENCR_3DES</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
</tbody>
</table>
Part C | ENCR_3DES | PRF_HMAC_SHA1 | AUTH_AES_XCBC_96 | AUTH_HMAC_SHA1_96 | Group 2
---|---|---|---|---|---

Part D | ENCR_3DES | PRF_HMAC_SHA1 | AUTH_HMAC_SHA1_96 | Group 14 or Group 24, Group 2

- Packet #3 CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proposal Length</td>
<td>44</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1 (IKE)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td># of Transforms</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>According to above configuration</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>2 (PRF)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>4 (D-H)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (1024 MODP Group)</td>
<td></td>
</tr>
</tbody>
</table>

Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type
IPv6 FORUM TECHNICAL DOCUMENT

Part A

6. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo Random Function (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type "REKEY_SA" and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithm (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type "REKEY_SA" and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

Part D: Multiple D-H Group (BASIC)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type "REKEY_SA" and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
24. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Part B

Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 10: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 12: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Part C**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Part D**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 22: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 24: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Possible Problems:**

- none
Test IKEv2.EN.R.1.2.6.6: Receiving Multiple Proposal

Purpose:
To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with multiple proposal to rekey IKE_SA.

References:
- [RFC 4306] - Sections 2.7, 2.8 and 3.3

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

TN1 transmits a CREATE_CHILD_SA request including a SA payload which contains the two proposals as follows:

<table>
<thead>
<tr>
<th>IKE_SA_INIT exchanges Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposals</td>
</tr>
<tr>
<td>Part A</td>
</tr>
<tr>
<td>Proposal #2</td>
</tr>
</tbody>
</table>

Packet #1 See Common Packet #1
Packet #2 See Common Packet #3
Packet #3 See below
### Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #11</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #11</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #11</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the common packet #11</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #11</td>
</tr>
</tbody>
</table>

#### Proposal #1 SA Proposal

<table>
<thead>
<tr>
<th>Proposal #</th>
<th>Protocol ID</th>
<th>SPI Size</th>
<th># of Transforms</th>
<th>Transform Type</th>
<th>Transform Length</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal 1</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>1 (ENCR)</td>
<td>8</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Proposal 2</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>2 (PRF)</td>
<td>8</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Proposal 2</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>3 (INTEG)</td>
<td>8</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Proposal 2</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>4 (D-H)</td>
<td>8</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

#### Proposal #2 SA Proposal

<table>
<thead>
<tr>
<th>Proposal #</th>
<th>Protocol ID</th>
<th>SPI Size</th>
<th># of Transforms</th>
<th>Transform Type</th>
<th>Transform Length</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal 1</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>1 (ENCR)</td>
<td>8</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Proposal 2</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>2 (PRF)</td>
<td>8</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Proposal 2</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>3 (INTEG)</td>
<td>8</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>Proposal 2</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>4 (D-H)</td>
<td>8</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo Random Function (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithms (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

Part D: Multiple D-H Group (BASIC)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.

24. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Part B**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 10: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 12: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Part C**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Part D**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 22: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 24: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Possible Problems:**

- none
Test IKEv2.EN.R.1.2.6.7: Changing PRFs when rekeying the IKE_SA

Purpose:

To verify an IKEv2 device properly uses new IKE_SA.

References:

- [RFC 4306] - Sections 2.8
- [RFC 4718] - Sections 5.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  Configure the devices according to the Common Configuration except for italic parameters.

<table>
<thead>
<tr>
<th>IKE_SA Rekeying Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption</td>
</tr>
<tr>
<td>Part A</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1 See Common Packet #1
Packet #3: CREATE_CHILD_SA request
Packet #3 is same as Common Packet #11 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform Type PRF is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td></td>
<td>8</td>
<td>2 (PRF)</td>
<td>0</td>
<td>4 (PRF_AES128_XCBC)</td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request to the NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits an INFORMATIONAL request with no payloads protected by the new IKE_SA and the Message ID field in the IKE header is zero.
8. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 14" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s responder’s SPI value in the SPI field.

Step 8: Judgment #4
The NUT responds with an INFORMATIONAL response with no payloads protected by the new IKE_SA and the Message ID field in the IKE header is zero.

Possible Problems:

- none
Test IKEv2.EN.R.1.2.6.8: D-H transform NONE when rekeying the IKE_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.EN.R.1.2.6.9: Rekeying Failure

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SAs with an unacceptable SA payload.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

- Packet #1: See Common Packet #1
- Packet #2: See Common Packet #3
- Packet #3: See below

<table>
<thead>
<tr>
<th>Packet #3: CREATE_CHILD_SA request</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
</tr>
<tr>
<td>UDP Header</td>
</tr>
<tr>
<td>IKEv2 Header</td>
</tr>
<tr>
<td>E Payload</td>
</tr>
<tr>
<td>N Payload</td>
</tr>
<tr>
<td>N Payload</td>
</tr>
<tr>
<td>Proposal #1</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
</tr>
<tr>
<td>SA Transform</td>
</tr>
<tr>
<td>SA Transform</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request to rekey the established IKE_SA to the NUT. The CREATE_CHILD_SA request includes a SA payload with a proposal unaccepted by the NUT.
6. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.
Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including a Notify payload of type NO_PROPOSAL_CHOSEN.

Possible Problems:

• None.
Group 2.7. Creating new CHILD_SAs Using a CREATE_CHILD_SA exchange

Test IKEv2.EN.R.1.2.7.1: Receipt of cryptographically valid message on the new SA

Purpose:
To verify an IKEv2 device properly handles CREATE_CHILD_SA to create a new CHILD_SA.

References:
- [RFC 4306] - Sections 2.8 and 2.18

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>ID1 Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #3</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT 513 IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Starting Address</td>
<td>TN1’s Global Address on Link A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ending Address</td>
<td>TN1’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Starting Address</td>
<td>NUT’s Global Address on Link X</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ending Address</td>
<td>NUT’s Global Address on Link X</td>
</tr>
</tbody>
</table>

- Packet #3: TCP SYN packet

IPv6 Header
- Source Address: TN1’s Global Address on Link X
- Destination Address: NUT’s Global Address on Link A

ESP
- Security Parameter Index: CHILD_SA’s SPI value used by this message
- Sequence Number: The value incremented the previous encrypted packet’s Sequence Number by one.
- Padding Data: Subsequent data encrypted by underlying encryption algorithm
- Padding: Any value which to be a multiple of the encryption block size
- Pad Length: The length of the Padding field
- Next Header: 6 (TCP)
- Integrity Check Value: The checksum must be valid by calculation according to the manner described in RFC

TCP Header
- Source Port: 30000
- Destination Port: 30000
- Flags: SYN (0x02)

- Packet #5: CREATE_CHILD_SA request

IPv6 Header
- Same as the Common Packet #7

UDP Header
- Same as the Common Packet #7

IKEv2 Header
- Same as the Common Packet #7

E Payload
- Same as the Common Packet #7

IDi Payload
- Same as the Common Packet #7

AUTH Payload
- Same as the Common Packet #7

N Payload
- Same as the Common Packet #7

SA Payload
- Same as the Common Packet #7

TSi Payload
- Other fields are same as the Common Packet #7
- Traffic Selectors: See below

TSr Payload
- Other fields are same as the Common Packet #7
- Traffic Selectors: See below
### IPv6-ICMP (IP Protocol ID 58)

<table>
<thead>
<tr>
<th>IP Protocol ID</th>
<th>58 (IPv6-ICMP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>Ending Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

#### Packet #6: TCP SYN packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
<td>CHILD_SA’s SPI value used by this message</td>
</tr>
<tr>
<td></td>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
</tr>
<tr>
<td></td>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Next Header</td>
<td>6 (TCP)</td>
</tr>
<tr>
<td></td>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
</tr>
<tr>
<td>TCP Header</td>
<td>Source Port</td>
<td>30000</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>30000</td>
</tr>
<tr>
<td></td>
<td>Flags</td>
<td>SYN (0x02)</td>
</tr>
</tbody>
</table>

### Part A: (ADVANCED)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a TCP-SYN packet with IPsec ESP using corresponding algorithms to closed port 30000 on NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a TCP-SYN packet with IPsec ESP using corresponding algorithms to closed port 30000 on NUT.
12. Observe the messages transmitted on Link A.
13. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
14. Observe the messages transmitted on Link A.

**Observable Results:**

### Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a TCP-RST packet with IPsec ESP using corresponding algorithms.
Step 8: Judgment #4
The NUT never transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 12: Judgment #6
The NUT transmits a TCP-RST packet with IPsec ESP using corresponding algorithms.

Step 14: Judgment #7
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- If the NUT uses TCP port 30000 for other applications, the TN1 transmits TCP-SYN packets to other closed TCP port on the NUT.
Group 2.8. Error Handling

Test IKEv2.EN.R.1.2.8.1: AUTHENTICATION_FAILED

This test case was deleted at revision 1.1.0.
Group 2.9. Non zero RESERVED fields

Test IKEv2.EN.R.1.2.9.1: Non zero RESERVED fields in CREATE_CHILD_SA request

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part</th>
<th>Packet #1</th>
<th>Packet #2</th>
<th>Packet #3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>See Common Packet #1</td>
<td>See Common Packet #3</td>
<td>See Common Packet #13</td>
</tr>
</tbody>
</table>

All RESERVED fields are set to one.

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

- none
Group 3. The INFORMATIONAL Exchange

Group 3.1. Header and Payload Formats

Test IKEv2.EN.R.1.3.1.1: Sending INFORMATIONAL response

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 1.1.2, 1.4, 3.1 and 3.14

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

Part A: IKE Header Format (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an INFORMATIONAL request with no payloads to the NUT.
6. Observe the messages transmitted on Link A.

Part B: Encrypted Payload Format (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN1 transmits an INFORMATIONAL request with no payloads to the NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an INFORMATIONAL response including properly formatted IKE Header containing following values:

- An IKE_SA Initiator’s SPI field is set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field is set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.

![Figure 82 Header format](image_url)
A Next Payload field is set to Encrypted Payload (46).
A Major Version field is set to 2.
A Minor Version field is set to zero.
An Exchange Type field is set to INFORMATIONAL (37).
A Flags field is set to (00000100)\(_2\) = (4)\(_{10}\).
A Message ID field is set to the same value as corresponding IKEv2 request message’s Message ID.
A Length field is set to the length of the message (header + payloads) in octets.

**Part B**

**Step 9: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 11: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 14: Judgment #3**
The NUT transmits an INFORMATIONAL response including properly formatted Encrypted Payload containing following values:

- A Next Payload field is set to zero.
- A Critical field is set to zero.
- A RESERVED field is set to zero.
- A Payload Length field is set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Check sum Data.
- An Initialization Vector field is set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
- An Encrypted IKE Payloads field is set to subsequent payloads encrypted by ENCR_3DES.
- A Padding field is set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
- A Pad Length field is set to the length of the Padding field.
- An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

Possible Problems:

- None.
Group 3.2. Use of Retransmission Timers

Test IKEv2.EN.R.1.3.2.1: Receipt of retransmitted INFORMATIONAL request

Purpose:

To verify an IKEv2 device properly handles the retransmission.

References:

- [RFC 4306] - Sections 1.1.2, 1.4 and 2.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 523 IPv6 Ready Logo Program IKEv2
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

Packet #2 See Common Packet #3
Packet #3 See Common Packet #17
Packet #4 See Common Packet #17
(same Message ID as packet #3)

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an INFORMATIONAL request with no payloads.
6. Observe the messages transmitted on Link A.
7. Observe the messages transmitted on Link A.
8. TN1 transmits an INFORMATIONAL request with no payloads. The Message ID is the same as step 5.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an INFORMATIONAL response followed by an Encrypted payload with no payloads contained in it.

Step 7: Judgment #4
The NUT never retransmits an INFORMATIONAL response followed by an Encrypted payload with no payloads contained in it.

Step 9: Judgment #5
The NUT transmits an INFORMATIONAL response followed by an Encrypted payload with no payloads contained in it.

Possible Problems:

- none
Group 3.3. Non zero RESERVED fields

Test IKEv2.EN.R.1.3.3.1: Non RESERVED fields in INFORMATIONAL request

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A: (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Packet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>See Common Packet #1</td>
</tr>
<tr>
<td>#2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>#3</td>
<td>See Common Packet #17</td>
</tr>
<tr>
<td></td>
<td>All RESERVED fields are set to one.</td>
</tr>
</tbody>
</table>
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an INFORMATIONAL request with no payloads. All RESERVED fields in the message are set to one.
6. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an INFOMATIONAL response followed by an Encrypted payload with no payloads contained in it.

Possible Problems:

- None
Group 4. RFC 5996

Group 4.1. Rekeying IKE SAs Using a CREATE_HLD_SA Exchange


Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 5996] - Section 2.18

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
</tbody>
</table>

![Packet Diagram]

IPv6 FORUM TECHNICAL DOCUMENT 528 IPv6 Ready Logo Program IKEv2
Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT. Observe the messages transmitted on Link A.
6. Observe the messages transmitted on Link A.
7. TN1 transmits a CREATE_CHILD_SA request including a SA payload. A proposal in the SA payload contains 1 (IKE) in the Protocol ID field, 8 in the SPI size field and the rekeyed IKE_SA’s initiator’s SPI value. The proposal has the value "NONE" for the Diffie-Hellman transform.
8. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 8: Judgment #4
The NUT transmits a CREATE_CHILD_SA response with Notify payload of type NO_PROPOSAL_CHOSEN.

Possible Problems:

- None
IPv6 FORUM TECHNICAL DOCUMENT

[EN.R.P86.L4030.ADD.1] Test IKEv2.EN.R.1.4.1.2.XXX: Sending
INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can handle a proposal that contains a Transform Type it does not understand

References:

- [RFC 5996] - Section 3.3.6

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request. The request has 2 proposals. One proposal has 5 Transforms which are "a Transform Type (240)", "ENC_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", and "D-H Group 2". Another proposal has 4 Transforms which are "ENC_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", and "D-H Group 2".
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Possible Problems:**

- None
**Purpose:**

To verify an IKEv2 device can handle a proposal that is missing a mandatory Transform Type

**References:**

- [RFC 5996] - Section 3.3.6

**Test Setup:**

- **Network Topology**
  Connect the devices according to the Common Topology.
- **Configuration**
  In each part, configure the devices according to the Common Configuration.
- **Pre-Sequence and Cleanup Sequence**
  IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKE_SA_INIT request has 2 SA Proposals.</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request. The request has 2 proposals. One proposal has 3 Transforms which "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", and "D-H Group 2". Another proposal has 4 Transforms which are "ENC_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", and "D-H Group 2".
2. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT request including "ENC_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Possible Problems:

- None
[EN.R.P86.L4034.ADD.1] Test IKEv2.EN.R.1.4.1.4.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can handle a transform that it does not understand

References:

- [RFC 5996] - Section 3.3.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKE_SA_INIT request has 5 SA Transforms.</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request. The request has 5 Transforms which are "1 (ENCR) as Transform Type and 1023 as Transform ID", "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2".
2. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Possible Problems:

- None
IPv6 FORUM TECHNICAL DOCUMENT

[EN.R.P86.L4034.ADD.2] Test IKEv2.EN.R.1.4.1.5.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can handle a Transform Attribute it does not understand

References:

- [RFC 5996] - Section 3.3.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

- IKE_SA_INIT request has 5 SA Transforms.

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request. The request has 5 Transforms which are "ENCR_3DES with Transform Attribute of type KeyLength and value 192", "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2"

2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1

The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Possible Problems:

- None
[EN.R.P57.L2663.ADD] Test IKEv2.EN.R.1.4.1.6.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can notify AUTHENTICATION_FAILED

References:

- [RFC 5996] - Section 2.21.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A: (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT. The response includes invalid Authentication Data 0x0123456789abcdef0123456789abcdef01234567.</td>
</tr>
<tr>
<td>4. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

Observable Results:

Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including Notify payload of type AUTHENTICATION_FAILED.

**Possible Problems:**

- None
[EN.R.P69.L3234.ADD] Test IKEv2.EN.R.1.4.1.7.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can process CHILD_CHILD_SA request to close a Child SA that it is currently rekeying.

References:

- [RFC 5996] - Section 2.25.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 539 IPv6 Ready Logo Program IKEv2
Packet #4: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Delete Payload</th>
<th>Next Payload</th>
<th>Critical</th>
<th>Reserved</th>
<th>Payload Length</th>
<th>Protocol ID</th>
<th>SPI Size</th>
<th># of SPIs</th>
<th>Security Parameter Index(es) (SPI)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>12</td>
<td>3 (ESP)</td>
<td>4</td>
<td>1</td>
<td>SPI negotiated by Initial Exchange</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT. Observe the messages transmitted on Link A.
6. Observe the messages transmitted on Link A.
7. Repeat Steps 5 and 6 until lifetime of SA is expired for 30 seconds.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request. The request includes a Delete payload with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Step 8: Judgment #4
The NUT transmits a CREATE_CHILD_SA request to rekey a Child SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence
Numbers” as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 10: Judgment #5**
The NUT transmits an INFORMATIONAL response. The response includes a Delete payload with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.

**Possible Problems:**

- None
Test IKEv2.EN.R.1.4.1.8.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can process INFORMATIONAL request to close IKE SA that it is currently rekeying.

References:

- [RFC 5996] - Section 2.25.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common below</td>
</tr>
</tbody>
</table>

- **Packet #4: INFORMATIONAL request**

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
</tbody>
</table>

**E Payload**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>16</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1 (IKE_SA)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td># of SPIs</td>
<td>0</td>
</tr>
<tr>
<td>Security Parameter Index(es) (SPI)</td>
<td>empty</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT. Observe the messages transmitted on Link A.
6. Observe the messages transmitted on Link A.
7. Repeat Steps 5 and 6 until lifetime of SA is expired for 30 seconds.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request. The request includes a Delete payload with 1 (IKE) as Protocol ID, zero as SPI Size and no SPI value.
10. Observe the messages transmitted on Link A.
11. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 8: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request to rekey IKE SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 10: Judgment #5**  
The NUT transmits an INFORMATIONAL response with no payloads.

**Step 11: Judgment #6**  
The NUT does not retransmit a CREATE_CHILD_SA request to rekey a Child SA.

**Possible Problems:**

- None

Purpose:

To verify an IKEv2 device can process CREATE_CHILD_SA request to rekey a Child SA when it is currently rekeying the IKE SA.

References:

- [RFC 5996] - Section 2.5.2

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 545 IPv6 Ready Logo Program IKEv2
**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT. Observe the messages transmitted on Link A.
6. Observe the messages transmitted on Link A.
7. Repeat Steps 5 and 6 until lifetime of SA is expired for 30 seconds.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request to rekey a Child SA. The request includes "ENC_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.
10. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENC_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENC_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 8: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request to rekey a Child SA. The message includes "ENC_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 10: Judgment #5**
The NUT transmits a CREATE_CHILD_SA response. The response includes a Notify payload of type TEMPORRAY_FAILURE.

**Possible Problems:**

- None
Purpose:

To verify an IKEv2 device can process CREATE_CHILD_SA request to delete a Child SA when it is currently rekeying the IKE SA.

References:

- [RFC 5996] - Section 2.5.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 547 IPv6 Ready Logo Program IKEv2
Packet #4: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Delete Payload</th>
<th>Next Payload</th>
<th>42 (Delete)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td># of SPIs</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Security Parameter Index(es) (SPI)</td>
<td>SPI negotiated by Initial Exchange</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT. Observe the messages transmitted on Link A.
6. Observe the messages transmitted on Link A.
7. Repeat Steps 5 and 6 until lifetime of SA is expired for 30 seconds.
8. Observe the messages transmitted on Link A.
9. TN1 transmits an INFORMATIONAL request. The request includes a Delete payload with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.
10. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 6: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 8: Judgment #4**
TN1 transmits an INFORMATIONAL request. The request includes a Delete payload with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.
Step 10: Judgment #5
The NUT transmits an INFORMATIONAL response with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.

Possible Problems:

- None
Section 1.2.2. Endpoint to Security Gateway Tunnel

Group 1. The Initial Exchanges
Group 1.1. Header and Payload Formats

Test IKEv2.EN.R.2.1.1.1: Sending IKE_AUTH response

Purpose:

To verify an IKEv2 device transmits IKE_AUTH response using properly Header and Payloads format

References:

- [RFC 4306] - Sections 1.2, 2.15, 3.1, 3.2, 3.3, 3.5, 3.8, 3.10, 3.13 and 3.14

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A: IKE Header Format (ADVANCED)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. TN1 transmits an IKE_SA_INIT request to NUT.</td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>3. TN1 transmits an IKE_SA_INIT request to NUT.</td>
</tr>
<tr>
<td>4. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part B: Encrypted Payload Format (ADVANCED)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. TN1 transmits an IKE_SA_INIT request to NUT.</td>
</tr>
<tr>
<td>6. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>7. TN1 transmits an IKE_SA_INIT request to NUT.</td>
</tr>
</tbody>
</table>
8. Observe the messages transmitted on Link A.

**Part C: IDr Payload Format (ADVANCED)**
9. TN1 transmits an IKE_SA_INIT request to NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an IKE_SA_INIT request to NUT.
12. Observe the messages transmitted on Link A.

**Part D: AUTH Payload Format (ADVANCED)**
13. TN1 transmits an IKE_SA_INIT request to NUT.
14. Observe the messages transmitted on Link A.
15. TN1 transmits an IKE_SA_INIT request to NUT.
16. Observe the messages transmitted on Link A.

**Part E: SA Payload Format (ADVANCED)**
17. TN1 transmits an IKE_SA_INIT request to NUT.
18. Observe the messages transmitted on Link A.
19. TN1 transmits an IKE_SA_INIT request to NUT.
20. Observe the messages transmitted on Link A.

**Part F: TSi Payload Format (ADVANCED)**
21. TN1 transmits an IKE_SA_INIT request to NUT.
22. Observe the messages transmitted on Link A.
23. TN1 transmits an IKE_SA_INIT request to NUT.
24. Observe the messages transmitted on Link A.

**Part G: TSr Payload Format (ADVANCED)**
25. TN1 transmits an IKE_SA_INIT request to NUT.
26. Observe the messages transmitted on Link A.
27. TN1 transmits an IKE_SA_INIT request to NUT.
28. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted IKE Header containing following values:
IPv6 Ready Logo Program IKEv2

Figure 84 Header format

- An IKE_SA Initiator’s SPI field set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field set to Encrypted Payload (46).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to IKE_AUTH (35).
- A Flags field set to (00010000)_2 = (16)_10.
- A Message ID field set to 1.
- A Length field set to the length of the message (header + payloads) in octets.

Part B

Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 8: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted Encrypted Payload containing following values:

Figure 85 Encrypted payload
• A Next Payload field set to IDr Payload (36).
• A Critical field set to zero.
• A RESERVED field set to zero.
• A Payload Length field set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Check sum Data.
• An Initialization Vector field set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm.
• An Encrypted IKE Payloads field set to encrypted IKE Payloads
• A Padding field set to any value which to be a multiple of the encryption block size.
• A Pad Length field set to the length of the Padding field.
• An Integrity Checksum Data set to the cryptographic checksum of the entire message. The checksum must be valid.

Part C

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted ID Payload containing following values:

![Figure 86 ID Payload format](image-url)

• A Next Payload field set to AUTH Payload (39).
• A Critical field set to zero.
• A RESERVED field set to zero.
• A Payload Length field set to length of the current payload.
• An ID Type field set to ID_IPV6_ADDR (5).
• A RESERVED field set to zero.
• An Identification Data field set to the NUT address.

Part D

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted AUTH Payload containing following values:

**Figure 87 AUTH Payload format**

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- An Auth Method field set to Shared Key Message Integrity Code (2).
- A RESERVED field set to zero.
- An Authentication Data field set to correct authentication value.

**Part E**

**Step 18: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field set to TSi Payload (44).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.

A Proposals field set to following.

- A 0 or 2 field set to zero (last).
- A RESREVD field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes.
- A Proposal # field set to 1.
- A Protocol ID field set to ESP (3).
- A SPI Size field set to 4.
- A # of Transforms field set to 3.
- A SPI field set to the sending entity’s SPI (4 octets value)

Transform field set to following (There are 3 Transform Structures).
Part F

Step 22: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 24: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted TSi Payload containing following values:

Figure 91 Transform sub-structure format

- A 0 or 3 field set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute.
- A Transform Type field set to ENCR (1).
- A RESERVED field set to zero.
- A Transform ID set to ENCR_3DES (3).

- A 0 or 3 field set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute.
- A Transform Type field set to INTEG (3).
- A RESERVED field set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

- A 0 or 3 field set to zero.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute.
- A Transform Type field set to ESN (5).
- A RESERVED field set to zero.
- A Transform ID set to No Extended Sequence Numbers (0).
Figure 92 TSi Payload format

- A Next Payload field set to TSr Payload (45).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to 1.
- A RESERVED field set to zero.

Traffic Selectors field set to following.

Figure 93 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to NUT address.
- A Ending Address field set to NUT address.

Part G

Step 26: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 28: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted TSr Payload containing following values:

- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to 1.
- A RESERVED field set to zero.

Traffic Selectors field set to following.

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to TN1 address.
- An Ending Address field set to TN1 address.

Possible Problems:

- None.
Test IKEv2.EN.R.2.1.1.2: Use of CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key.

References:

- [RFC 4306] - Sections 1.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet</th>
<th>See Common Packet #1</th>
<th>Packet</th>
<th>See Common Packet #5</th>
<th>Packet</th>
<th>See Common Packet #20</th>
</tr>
</thead>
</table>

Part A (ADVANCED)
1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT response to NUT.
4. Observe the messages transmitted on Link A.
5. TH1 transmits an Echo Request and TN1 forwards an Echo Request with IPsec ESP using corresponding algorithms to NUT.
6. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Section 2. Security Gateway
Section 2.1. Initiator
Section 2.1.1. Security Gateway to Security Gateway Tunnel
Group 1. The Initial Exchanges
Group 1.1. Header and Payload Formats

Test IKEv2.SGW.I.1.1.1.1: Sending IKE_SA_INIT request

Purpose:
To verify an IKEv2 device transmits IKE_SA_INIT request using properly Header and Payloads format

References:
- [RFC4306] - Section 1.2, 2.10, 3.1, 3.2, 3.3, 3.4 and 3.9
- [RFC 4718] - Sections 7.4

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A: IKE Header Format (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part B: SA Payload Format (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>4. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part C: KE Payload Format (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>6. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part D: Nonce Payload Format (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>8. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

Observable Results:
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including properly formatted IKE Header containing following values:

- An IKE_SA Initiator’s SPI field set to a 64-bits value chosen by the NUT. It MUST not be zero.
- An IKE_SA Responder’s SPI field set to zero.
- A Next Payload field set to SA Payload (33).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to IKE_SA_INIT (34).
- A Flags field set to (00010000)₂ = (16)₁₀.
- A Message ID field set to zero.
- A Length field set to the length of the message (header + payloads) in octets.

Part B
Step 4: Judgment #1
The NUT transmits an IKE_SA_INIT request including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field set to KE Payload (34).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.

The following proposal must be included in Proposals field.
Proposal #1
- A 0 or 2 field set to zero if this structure is the last proposal, otherwise set to 2.
- A RESERVD field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes. It is 40 bytes for this proposal according to Common Configuration.
- A Proposal # field set to 1 if this structure is the first proposal, otherwise set to 1 greater than the previous proposal.
- A Protocol ID field set to IKE (1).
- A SPI Size field set to zero.
- A # of Transforms field set to 4.

A Transform field set to following (There are 4 Transform Structures).

Transform #1
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field set to ENCR (1).
- A RESERVED field set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for PRF_HMAC_SHA1.
- A Transform Type field set to PRF (2).
- A RESERVED field set to zero.
- A Transform ID set to PRF_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field set to INTEG (3).
- A RESERVED field set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #4
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for 1024 MODP Group.
- A Transform Type field set to D-H (4).
- A RESERVED field set to zero.
- A Transform ID set to Group2 (2).

**Part C**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including properly formatted KE Payload containing following values:

![Figure 101 KE Payload format](image)

- A Next Payload field set to Nonce Payload (40).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 136 bytes for Group 2.
- A DH Group field set to Group2 (2).
- A RESERVED field set to zero.
- A Key Exchange Data field set to Diffie-Hellman public value. The length of the Key Exchange Data field must be equal to 1024 bit.
Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT request including properly formatted Nonce Payload containing following values:

- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Nonce Data field set to random data generated by the transmitting entity. The size of the Nonce must between 16 and 256 octets.

Possible Problems:

- IKE_SA_INIT request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- The implementation may not set single proposal by the implementation policy. In this case, Security Association Payload contains multiple proposals.

- Each of transforms can be located in the any order.
Test IKEv2.SGW.I.1.1.1.2: Sending IKE_AUTH request

Purpose:

To verify an IKEv2 device transmits IKE_AUTH request using properly Header and Payloads format.

References:

- [RFC 4306] - Sections 1.2, 2.15, 3.1, 3.2, 3.3, 3.5, 3.8, 3.10, 3.13 and 3.14

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A: IKE Header Format (BASIC)</th>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. TN1 responds with an IKE_SA_INIT response to the NUT.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Observe the messages transmitted on Link A.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part B: Encrypted Payload Format (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>6. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>7. TN1 responds with an IKE_SA_INIT response to the NUT.</td>
</tr>
<tr>
<td>8. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part C: IDi Payload Format (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>10. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>11. TN1 responds with an IKE_SA_INIT response to the NUT.</td>
</tr>
<tr>
<td>12. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>
Part D: AUTH Payload Format (BASIC)
13. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. TN1 responds with an IKE_SA_INIT response to the NUT.
16. Observe the messages transmitted on Link A.

Part E: SA Payload Format (BASIC)
17. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. TN1 responds with an IKE_SA_INIT response to the NUT.
20. Observe the messages transmitted on Link A.

Part F: TSi Payload Format (BASIC)
21. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
22. Observe the messages transmitted on Link A.
23. TN1 responds with an IKE_SA_INIT response to the NUT.
24. Observe the messages transmitted on Link A.

Part G: TSr Payload Format (BASIC)
25. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. TN1 responds with an IKE_SA_INIT response to the NUT.
28. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted IKE Header containing following values:

![Figure 103 Header format](image-url)
Initiator’s SPI field value.

- An IKE_SA Responder’s SPI field set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field set to Encrypted Payload (46).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to IKE_AUTH (35).
- A Flags field set to (00010000)\textsubscript{2} = (16)\textsubscript{10}.
- A Message ID field set to 1.
- A Length field set to the length of the message (header + payloads) in octets.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted Encrypted Payload containing following values:

### Figure 104 Encrypted payload

- A Next Payload field set to IDi Payload (35).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Check sum Data.
- An Initialization Vector field set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
- An Encrypted IKE Payloads field set to subsequent payloads encrypted by ENCR_3DES.
- A Padding field set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
- A Pad Length field set to the length of the Padding field.
- An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum
must be valid by calculation according to the manner described in RFC.

Part C

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted ID Payload containing following values:

![Figure 105 ID Payload format](image)

- A Next Payload field set to AUTH Payload (39).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 24 bytes for ID_IPV6_ADDR.
- An ID Type field set to ID_IPV6_ADDR (5).
- A RESERVED field set to zero.
- An Identification Data field set to the NUT address.

Part D

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted AUTH Payload containing following values:
Figure 106 AUTH Payload format

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 28 bytes for PRF_HMAC_SHA1.
- An Auth Method field set to Shared Key Message Integrity Code (2).
- A RESERVED field set to zero.
- An Authentication Data field set to correct authentication value according to the manner described in RFC. It is 160 bytes length in PRF_HMAC_SHA1 case.

Part E

Step 18: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 20: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field set to TSi Payload (44).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.

The following proposal must be included in Proposals field:

The following proposal must be included in Proposals field.

- A 0 or 2 field set to zero if this structure is the last proposal, otherwise set to 2.
- A RESREVD field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field set to 1 if this structure is the first proposal, otherwise set to 1 greater than the previous proposal.
- A Protocol ID field set to ESP (3).
- A SPI Size field set to 4.
- A # of Transforms field set to 3.
- A SPI field set to the sending entity’s SPI (4 octets value)

Transform field set to following (There are 3 Transform Structures).
Transform #1
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field set to ENCR (1).
- A RESERVED field set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field set to INTEG (3).
- A RESERVED field set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
- A Transform Type field set to ESN (5).
- A RESERVED field set to zero.
- A Transform ID set to No Extended Sequence Numbers (0).

Part F

Step 22: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 24: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted TSi Payload containing following values:
Figure 111 TSi Payload format

- A Next Payload field set to TSr Payload (45).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to the number of actual traffic selectors.
- A RESERVED field set to zero.

The following traffic selector must be included in Traffic Selectors field.

Figure 112 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to Prefix B.
- A Ending Address field set to greater than or equal to Prefix B.

**Part G**

**Step 26: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 28: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted TSr Payload containing following values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Next Payload field</td>
<td>set to zero</td>
</tr>
<tr>
<td>A Critical field</td>
<td>set to zero</td>
</tr>
<tr>
<td>A RESERVED field</td>
<td>set to zero</td>
</tr>
<tr>
<td>A Payload Length field</td>
<td>set to length of the current payload</td>
</tr>
<tr>
<td>A Number of TSs field</td>
<td>set to the number of actual traffic selectors</td>
</tr>
<tr>
<td>A RESERVED field</td>
<td>set to zero</td>
</tr>
</tbody>
</table>

The following traffic selector must be included in Traffic Selectors field.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A TS Type</td>
<td>set to TS_IPV6_ADDR_RANGE (8)</td>
</tr>
<tr>
<td>An IP Protocol ID field</td>
<td>set to zero</td>
</tr>
<tr>
<td>A Selector Length field</td>
<td>set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.</td>
</tr>
<tr>
<td>A Start Port field</td>
<td>set to zero</td>
</tr>
<tr>
<td>An End Port field</td>
<td>set to 65535</td>
</tr>
<tr>
<td>A Starting Address field</td>
<td>set to less than or equal to Prefix Y</td>
</tr>
<tr>
<td>An Ending Address field</td>
<td>set to less than or equal to Prefix Y</td>
</tr>
</tbody>
</table>

Possible Problems:

- IKE_AUTH request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload
may be different from this sample.

- The implementation may not set single proposal by the implementation policy. In this case, Security Association Payload contains multiple proposals.

- Each of transforms can be located in the any order.

- The implementation may not set single traffic selector by the implementation policy. In this case, Traffic Selector Payload contains multiple proposals.
Test IKEv2.SGW.I.1.1.1.3: Use of CHILD_SA

Purpose:
To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key.

References:
- [RFC 4306] - Sections 1.2

Test Setup:
- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #1</td>
<td>See Common Packet #2</td>
</tr>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Possible Problems:

- Because the destination address of Echo Request is the TN itself, TN may respond to Echo Request automatically. In that case, TH2 can send Echo Reply to TH1 instead of sending Echo Request.
Group 1.2. Use of Retransmission Timers

Test IKEv2.SGW.I.1.1.2.1: Retransmissions of IKE_SA_INIT requests

Purpose:
To verify an IKEv2 device retransmits IKE_SA_INIT request using properly Header and Payloads format

References:
- [RFC 4306] - Sections 2.1, 2.2 and 2.4
- [RFC 4718] - Sections 2.2 and 2.3

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set retransmission timer to 1 second.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 waits for the event of a timeout on NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 4: Judgment #2
The NUT retransmits an IKE_SA_INIT request which has the same Message ID value as the previous IKE_SA_INIT request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different retransmission timers. If it is impossible to configure the retransmission timer, modifying tester is required.
Test IKEv2.SGW.I.1.1.2.2: Stop of retransmission of IKE_SA_INIT requests

Purpose:

To verify an IKEv2 device stops retransmission when it receives the corresponding response.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4
- [RFC 4718] - Sections 2.2 and 2.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set retransmission timer to 1 second.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
</table>

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 waits for the event of a timeout on NUT.
4. Observe the messages transmitted on Link A
5. TN1 responds with an IKE_SA_INIT response to the NUT.
6. TN1 waits for the event of a timeout on NUT.
7. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT retransmits an IKE_SA_INIT request which has the same Message ID value as the previous IKE_SA_INIT request’s Message ID value in IKE Header.

Step 7: Judgment #3
The NUT never retransmits an IKE_SA_INIT request which has the same Message ID value as the previous IKE_SA_INIT request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different retransmission timers. If it is imposibble to configure the retransmission timer, modifying tester is required.
Test IKEv2.SGW.I.1.1.2.3: Retransmissions of IKE_AUTH requests

Purpose:

To verify an IKEv2 device retransmits IKE_AUTH request using properly Header and Payloads format

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set retransmission timer to 1 second.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1  See Common Packet #2

Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 waits for the event of a timeout on NUT.
6. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES","PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 6: Judgment #3**
The NUT retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

**Possible Problems:**
- Each NUT has the different retransmission timers. If it is imposible to configure the retransmission timer, modifying tester is required.
Test IKEv2.SGW.I.1.1.2.4: Stop of retransmission of IKE_AUTH requests

Purpose:

To verify an IKEv2 device stops retransmission when it receives the corresponding response.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- **Network Topology**
  - Connect the devices according to the Common Topology.
- **Configuration**
  - In each part, configure the devices according to the Common Configuration. In addition, set retransmission timer to 1 second.
- **Pre-Sequence and Cleanup Sequence**
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 waits for the event of a timeout on NUT.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_AUTH response to the NUT.
8. TN1 waits for the event of a timeout on NUT.
9. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 6: Judgment #3**
The NUT retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

**Step 9: Judgment #4**
The NUT never retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

**Possible Problems:**

- Each NUT has the different retransmission timers. If it is impossible to configure the retransmission timer, modifying tester is required.
Group 1.3. State Synchronization and Connection Timeouts

Test IKEv2.SGW.I.1.1.3.1: State Synchronization with ICMP messages

Purpose:

To verify an IKEv2 device synchronizes its state when it receives ICMP messages.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #5: ICMPv6 Destination Unreachable

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TR1's Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT's Global Address on Link A</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Code</td>
<td>0</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. After reception of an Echo Reply via NUT, TR1 transmits ICMP Destination Unreachable Message to the NUT and then TH2 transmits an Echo Request to the TH1.
11. Observe the messages transmitted on Link B.
12. TH1 transmits an Echo Reply to TH2.
13. Observe the messages transmitted on Link A.

**Observable Results:**

- **Step 2: Judgment #1**
  The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

- **Step 4: Judgment #2**
  The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

- **Step 7: Judgment #3**
  The NUT forwards an Echo Request.

- **Step 9: Judgment #4**
  The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

- **Step 11: Judgment #5**
  The NUT forwards an Echo Request.

- **Step 13: Judgment #6**
  The NUT forwards an Echo Request.
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.3.2: State Synchronization with IKE messages

Purpose:

To verify an IKEv2 device synchronizes its state when it receives IKE messages.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- **Network Topology**
  Connect the devices according to the Common Topology.
- **Configuration**
  In each part, configure the devices according to the Common Configuration.
- **Pre-Sequence and Cleanup Sequence**
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #1</td>
<td></td>
</tr>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See Common Packet #21</td>
</tr>
</tbody>
</table>
Packet #4: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link X</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0–2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6–7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type</td>
<td>11 (INVALID_SPI)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. TN1 transmits INFORMATIONAL request with a Notify payload of type INVALID_SPI to the NUT.
11. TH2 transmits an Echo Request to TH1.
12. Observe the messages transmitted on Link B.
13. TH1 transmits an Echo Reply to TH2.
14. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 12: Judgment #5**
The NUT forwards an Echo Request.

**Step 14: Judgment #6**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Possible Problems:**

- None
Test IKEv2.SGW.I.1.1.3.3: Close connections when repeated attempts fail

Purpose:

To verify an IKEv2 device stops retransmission when it receives the corresponding response.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (BASIC)

9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. TN1 responds with an IKE_SA_INIT response to the NUT.
12. Observe the messages transmitted on Link A.
13. TN1 waits for the event of a timeout on the NUT.
14. Observe the messages transmitted on Link A.
15. Repeat Step 5 and Step 6 until the NUT’s last retransmission comes.
16. Observe the messages transmitted on Link A.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

Step 8: Judgment #4
The NUT never retransmits an IKE_AUTH request which has the same Message ID value as the previous IKE_AUTH request’s Message ID value in IKE Header.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.3.4: Close connections when receiving INITIAL_CONTACT

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.1.3.5: Sending Liveness check

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.1.3.6: Sending Delete Payload for IKE_SA

Purpose:

To verify an IKEv2 device transmits a Delete Payload, when IKE_SA is deleted.

References:

- [RFC 4306] - Sections 2.4 and 3.11

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 waits until expiring IKE_SA’s lifetime and does not respond to an INFORMATIONAL request with an INFORMATIONAL response for liveness check.
7. Observe the messages transmitted on Link B.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an INFORMATIONAL request with a Delete Payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.

**Possible Problems:**

- At Step 7, NUT can transmit INFORMATIONAL request with a Delete Payload including 2 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete CHILD_SA before transmitting an INFORMATIONAL request to delete IKE_SA.
Test IKEv2.SGW.I.1.1.3.7: Sending Delete Payload for CHILD_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.1.3.8: Sending Liveness check with unprotected messages

This test case was deleted at revision 1.1.0.
Group 1.4. Version Numbers and Forward Compatibility

Test IKEv2.SGW.I.1.1.4.1: Unrecognized payload types and critical bit is not set

Purpose:

To verify an IKEv2 device ignores invalid payload types when the invalid type payload’s critical bit is not set.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1 | See Common Packet #2
--- | ---
Packet #2 | See Common Packet #6
Packet #3 | See Common Packet #21
Packet #4 | See Common Packet #25
Packet #5 | See below
Packet #6 | See Common Packet #21
Packet #7 | See Common Packet #25

Packet #5: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>All fields are same as Common Packet #16 Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #16 Payload</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #16 Payload</td>
</tr>
<tr>
<td>E payload</td>
<td>Invalid payload type value</td>
</tr>
<tr>
<td></td>
<td>Other fields are same as Common Packet #16</td>
</tr>
</tbody>
</table>

Invalid Payload

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>33 (SA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>4</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

SA Payload
All fields are same as Common Packet #16 Payload

Ni, Nr payload
All fields are same as Common Packet #16 Payload

TSl Payload
All fields are same as Common Packet #16 Payload

TSr Payload
All fields are same as Common Packet #16 Payload

Part A: Invalid payload type 1 (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 1 and the invalid payload’s critical flag is not set.
13. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
14. Observe the messages transmitted on Link B.
15. TH1 transmits an Echo Response to the TH2.
16. Observe the messages transmitted on Link A.

Part B: Invalid payload type 32 (BASIC)

17. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. TN1 responds with an IKE_SA_INIT response to the NUT.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
22. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
23. Observe the messages transmitted on Link B.
24. TH1 transmits an Echo Reply to TH2.
25. Observe the messages transmitted on Link A.
26. Repeat Steps 22 through 25 until lifetime of SA is expired.
27. Observe the messages transmitted on Link A.
28. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 32 and the invalid payload’s critical flag is not set.
29. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
30. Observe the messages transmitted on Link B.
31. TH1 transmits an Echo Response to the TH2.
32. Observe the messages transmitted on Link A.

Part C: Invalid payload type 49 (BASIC)
33. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
34. Observe the messages transmitted on Link A.
35. TN1 responds with an IKE_SA_INIT response to the NUT.
36. Observe the messages transmitted on Link A.
37. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
38. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
39. Observe the messages transmitted on Link B.
40. TH1 transmits an Echo Reply to TH2.
41. Observe the messages transmitted on Link A.
42. Repeat Steps 38 through 41 until lifetime of SA is expired.
43. Observe the messages transmitted on Link A.
44. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 49 and the invalid payload’s critical flag is not set.
45. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
46. Observe the messages transmitted on Link B.
47. TH1 transmits an Echo Response to the TH2.
48. Observe the messages transmitted on Link A.

Part D: Invalid payload type 255 (BASIC)
49. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
50. Observe the messages transmitted on Link A.
51. TN1 responds with an IKE_SA_INIT response to the NUT.
52. Observe the messages transmitted on Link A.
53. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
54. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
55. Observe the messages transmitted on Link B.
56. TH1 transmits an Echo Reply to TH2.
57. Observe the messages transmitted on Link A.
58. Repeat Steps 54 through 57 until lifetime of SA is expired.
59. Observe the messages transmitted on Link A.
60. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 255 and the invalid payload’s critical flag is not set.
61. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
62. Observe the messages transmitted on Link B.
63. TH1 transmits an Echo Response to the TH2.
64. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 14: Judgment #6**
The NUT forwards an Echo Request to the TH1.

**Step 16: Judgment #7**
The NUT forwards an Echo Reply with IPsec ESP using the second negotiated algorithms.

*Part B*

**Step 18: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 23: Judgment #3**
The NUT forwards an Echo Request.

**Step 25: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

**Step 27: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 30: Judgment #6**
The NUT forwards an Echo Request to the TH1.

**Step 32: Judgment #7**
The NUT forwards an Echo Reply with IPsec ESP using the second negotiated algorithms.
**Part C**

**Step 34: Judgment #1**  
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 36: Judgment #2**  
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 39: Judgment #3**  
The NUT forwards an Echo Request.

**Step 41: Judgment #4**  
The NUT forwards an Echo Reply with IPsec ESP using the first negotitated algorithms.

**Step 43: Judgment #5**  
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 46: Judgment #6**  
The NUT forwards an Echo Request to the TH1.

**Step 48: Judgment #7**  
The NUT forwards an Echo Reply with IPsec ESP using the second negotitated algorithms.

---

**Part D**

**Step 50: Judgment #1**  
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 52: Judgment #2**  
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 55: Judgment #3**  
The NUT forwards an Echo Request.

**Step 57: Judgment #4**  
The NUT forwards an Echo Reply with IPsec ESP using the first negotitated algorithms.

**Step 59: Judgment #5**  
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 62: Judgment #6**  
The NUT forwards an Echo Request to the TH1.
Step 64: Judgment #7
The NUT forwards an Echo Reply with IPsec ESP using the second negotiated algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.4.2: Unrecognized payload types and critical bit is set

Purpose:

To verify an IKEv2 device rejects the messages with invalid payload types when the invalid type payload’s critical bit is set.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1 See Common Packet #2
Packet #2 See Common Packet #6
Packet #3 See Common Packet #21
Packet #4 See Common Packet #25
Packet #5 See below
Packet #6 See Common Packet #21
This packet is cryptographically protected by the CHILD_SA negotiated at Step 11.
Packet #7 See Common Packet #25

Packet #5: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>All fields are same as Common Packet #16 Payload</td>
</tr>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #16 Payload</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #16 Payload</td>
</tr>
<tr>
<td>E payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Payload</td>
<td>Other fields are same as Common Packet #16</td>
</tr>
<tr>
<td>Invalid Payload</td>
<td>Next Payload 33 (SA)</td>
</tr>
<tr>
<td>Critical</td>
<td>1</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>4</td>
</tr>
<tr>
<td>SA Payload</td>
<td>All fields are same as Common Packet #16 Payload</td>
</tr>
<tr>
<td>Ni, Nr payload</td>
<td>All fields are same as Common Packet #16 Payload</td>
</tr>
</tbody>
</table>
Part A: Invalid payload type 1 and Critical bit is set (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 1 and the invalid payload’s critical flag is set.
13. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
14. Observe the messages transmitted on Link B.
15. TH1 transmits an Echo Response to the TH2.
16. Observe the messages transmitted on Link A.

Part B: Invalid payload type 32 and Critical bit is set (BASIC)

17. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. TN1 responds with an IKE_SA_INIT response to the NUT.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
22. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
23. Observe the messages transmitted on Link B.
24. TH1 transmits an Echo Reply to TH2.
25. Observe the messages transmitted on Link A.
26. Repeat Steps 22 through 25 until lifetime of SA is expired.
27. Observe the messages transmitted on Link A.
28. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 32 and the invalid payload’s critical flag is set.
29. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
30. Observe the messages transmitted on Link B.
31. TH1 transmits an Echo Response to the TH2.
32. Observe the messages transmitted on Link A.

Part C: Invalid payload type 49 Critical bit is set (BASIC)
Part D: Invalid payload type 255 Critical bit is set (BASIC)

49. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
50. Observe the messages transmitted on Link A.
51. TN1 responds with an IKE_SA_INIT response to the NUT.
52. Observe the messages transmitted on Link A.
53. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
54. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
55. Observe the messages transmitted on Link B.
56. TH1 transmits an Echo Reply to TH2.
57. Observe the messages transmitted on Link A.
58. Repeat Steps 54 through 57 until lifetime of SA is expired.
59. Observe the messages transmitted on Link A.
60. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response which includes a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 255 and the invalid payload’s critical flag is set.
61. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
62. Observe the messages transmitted on Link B.
63. TH1 transmits an Echo Response to the TH2.
64. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotitated algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 14: Judgment #6
The NUT never forwards an Echo Request to the TH1.

Step 16: Judgment #7
The NUT never forwards an Echo Reply with IPsec ESP using the second negotitated algorithms.

Part B

Step 18: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 20: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 23: Judgment #3
The NUT forwards an Echo Request.

Step 25: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotitated algorithms.

Step 27: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 30: Judgment #6
The NUT never forwards an Echo Request to the TH1.

Step 32: Judgment #7
The NUT never forwards an Echo Reply with IPsec ESP using the second negotitated algorithms.

**Part C**

**Step 34: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 36: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 39: Judgment #3**
The NUT forwards an Echo Request.

**Step 41: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotitated algorithms.

**Step 43: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 46: Judgment #6**
The NUT never forwards an Echo Request to the TH1.

**Step 48: Judgment #7**
The NUT never forwards an Echo Reply with IPsec ESP using the second negotitated algorithms.

**Part D**

**Step 50: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 52: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 55: Judgment #3**
The NUT forwards an Echo Request.

**Step 57: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotitated algorithms.

**Step 59: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.
Step 62: Judgment #6
The NUT never forwards an Echo Request to the TH1.

Step 64: Judgment #7
The NUT never forwards an Echo Reply with IPsec ESP using the second negotiated algorithms.

Possible Problems:

- None.
Group 1.5. Cookies

Test IKEv2.SGW.I.1.1.5.1: Retrying IKE_SA_INIT request with a Notify payload of type COOKIE

Purpose:

To verify an IKEv2 device retries IKE_SA_INIT request using a Notify payload of type COOKIE.

References:

- [RFC 4306] - Sections 2.6 and 3.10.1
- [RFC 4718] - Sections 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>All fields are same as Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #2</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IKE_SA Initiator’s SPI</th>
<th>The same value as corresponding request’s IKE_SA_Initiator’s SPI value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKE_SA Responder’s SPI</td>
<td>0</td>
</tr>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td>Exchange Type</td>
<td>34 (IKE_SA_INIT)</td>
</tr>
<tr>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
</tbody>
</table>

Packet #1 See below
Part A: (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response including a Notify payload of type COOKIE to the NUT.
4. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_SA_INIT request including a Notify payload of type COOKIE containing following values:

![Figure 115 Notify Payload format](image-url)

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A SPI Size field set to zero.
- A Notify Message Type field set to COOKIE (16390).
- A Notification Data field set to the TN1 supplied cookie data.
Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.5.2: Interaction of COOKIE and INVALID_KE_PAYLOAD

Purpose:

To verify an IKEv2 device properly handles a series of the Initial Exchanges using a Notify payload of type COOKIE and type INVALID_KE_PAYLOAD.

References:

- [RFC 4306] - Sections 2.6, 2.7 and 3.10.1
- [RFC 4718] - Sections 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, configure the IKE_SA parameters as described as following. KEi payload must carry either D-H Group 14 public key value or D-H Group 24 public key value.

<table>
<thead>
<tr>
<th>Part</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2, Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

IKE_SA_INIT request (HDR, N COOKIE, SAI1, KEi, Ni) -> Sequence #1
IKE_SA_INIT request (HDR, SAI1, KEi, Ni)  -> Sequence #2
IKE_SA_INIT request (HDR, N COOKIE, SAI1, KEi, Ni)  -> Sequence #3
Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the common packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
</tbody>
</table>

Packet #1 See below
Packet #2 See below
Packet #3 See below
Packet #4 See Common Packet #2
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

Payload Length | Any
Protocol ID     | 0
SPI Size        | 0
Notify Message Type | COOKIE (16390)
Notification Data     | Cookie value

Packet #2: IKE_SA_INIT response

IPv6 Header | Same as the common packet #1
UDP Header  | Same as the common packet #1
IKEv2 Header | Other Fields are same as the common packet #1

<table>
<thead>
<tr>
<th>N Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
</tr>
<tr>
<td>Critical</td>
</tr>
<tr>
<td>Reserved</td>
</tr>
<tr>
<td>Payload Length</td>
</tr>
<tr>
<td>Protocol ID</td>
</tr>
<tr>
<td>SPI Size</td>
</tr>
<tr>
<td>Notify Message Type</td>
</tr>
<tr>
<td>Notification Data</td>
</tr>
</tbody>
</table>

Packet #3: IKE_SA_INIT response

IPv6 Header | Same as the common packet #1
UDP Header  | Same as the common packet #1
IKEv2 Header | Other Fields are same as the common packet #1

<table>
<thead>
<tr>
<th>N Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
</tr>
<tr>
<td>Critical</td>
</tr>
<tr>
<td>Reserved</td>
</tr>
<tr>
<td>Payload Length</td>
</tr>
<tr>
<td>Protocol ID</td>
</tr>
<tr>
<td>SPI Size</td>
</tr>
<tr>
<td>Notify Message Type</td>
</tr>
<tr>
<td>Notification Data</td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response including a Notify payload of type COOKIE to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 responds with an IKE_SA_INIT response including a Notify payload of type INVALID_KE_PAYLOAD to the NUT.
6. Observe the messages transmitted on Link A.
7. If the IKE_SA_INIT request from NUT includes a Notify payload of type COOKIE, TN1 responds with an IKE_SA_INIT response. The message has a different cookie value from the cookie value at Step3.
   A) Observe the messages transmitted on Link A.
   B) TN1 responds with an IKE_SA_INIT response.
8. If the IKE_SA_INIT request from NUT does not include a Notify payload of type COOKIE, TN1 responds with an IKE_SA_INIT response.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as
proposed algorithms. KEi payload has D-H Group 14 public key value. Depending on configuration, it is possible to use D-H Group 24 for SA proposal and KEi payload instead of D-H Group 14.

**Step 4: Judgment #2**
The NUT transmits an IKE_SA_INIT request. The message has a Notify payload of type COOKIE with the cookie data supplied by the responder as the first payload. All other payloads are unchanged.

**Step 6: Judgment #3**
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. The message can have a Notify payload of type COOKIE with the cookie data supplied by the responder at Step 5. All other payloads are unchanged.

**Step 7A: Judgment #4**
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. The message must have a Notify payload of type COOKIE with the cookie data supplied by the responder at Step 7. All other payloads are unchanged.

**Step 9: Judgment #5**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- None.
Test IKEv2.SGW.I.1.1.5.3: Interaction of COOKIE and INVALID_KE PAYLOAD with unoptimized Responder

Purpose:

To verify an IKEv2 device properly handles a series of the Initial Exchanges using a Notify payload of type COOKIE and type INVALID_KE PAYLOAD.

References:

- [RFC 4306] - Sections 2.6, 2.7 and 3.10.1
- [RFC 4718] - Sections 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, configure the IKE_SA parameters as described as following. KEi payload must carry either D-H Group 14 public key value or D-H Group 24 public key value.

<table>
<thead>
<tr>
<th>IKE_SA Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the common packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>41 (N)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>N Payload</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>0 (No Next Payload)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>Any</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>COOKIE (16390)</td>
</tr>
<tr>
<td>Notification Data</td>
<td>Cookie value</td>
</tr>
</tbody>
</table>

Packet #2: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the common packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>41 (N)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>N Payload</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>0 (No Next Payload)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>10</td>
</tr>
</tbody>
</table>

Packet #1 See below
Packet #2 See below
Packet #3 See below
Packet #4 See Common Packet #2
Packet #3: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the common packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the common packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the common packet #1</td>
</tr>
</tbody>
</table>

N Payload

- Next Payload: 41 (N)
- Critical: 0
- Payload Length: Any
- Protocol ID: 0
- SPI Size: 0
- Notify Message Type: COOKIE (16390)
- Notification Data: Different cookie value from Packet #1’s cookie value.

Part A: (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response including a Notify payload of type COOKIE to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 responds with an IKE_SA_INIT response including a Notify payload of type INVALID_KE_PAYLOAD to the NUT.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_SA_INIT response. The message has a different cookie value from the cookie value at Step3.
8. Observe the messages transmitted on Link A.
9. TN1 responds with an IKE_SA_INIT response.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload has D-H Group 14 public key value. Depending on configuration, it is possible to use D-H Group 24 for SA proposal and KEi payload instead of D-H Group 14.

**Step 4: Judgment #2**
The NUT transmits an IKE_SA_INIT request. The message has a Notify payload of type COOKIE with the cookie data supplied by the responder as the first payload. All other payloads are unchanged.

**Step 6: Judgment #3**
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. The message can have a Notify payload of type COOKIE with the cookie data supplied by the responder at Step 5.

**Step 8: Judgment #4**
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. The message must have a Notify payload of type COOKIE with the cookie data supplied by the responder at Step 7. All other payloads are unchanged.

Step 10: Judgment #5
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

- None.
Group 1.6. Cryptographic Algorithm Negotiation

Test IKEv2.SGW.I.1.1.6.1: Cryptographic Algorithm Negotiation for IKE_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-Shared key.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  From part A to part H, configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td><strong>ENCR_AES_CBC</strong></td>
<td><strong>PRF_HMAC_SHA1</strong></td>
<td><strong>AUTH_HMAC_SHA1_96</strong></td>
</tr>
<tr>
<td>Part B</td>
<td><strong>DELETED</strong></td>
<td><strong>DELETED</strong></td>
<td><strong>DELETED</strong></td>
</tr>
<tr>
<td>Part C</td>
<td><strong>ENCR_3DES</strong></td>
<td><strong>PRF_AES128_CBC</strong></td>
<td><strong>AUTH_HMAC_SHA1_96</strong></td>
</tr>
<tr>
<td>Part D</td>
<td><strong>ENCR_3DES</strong></td>
<td><strong>PRF_HMAC_SHA1</strong></td>
<td><strong>AUTH_AES_XCBC_96</strong></td>
</tr>
<tr>
<td>Part E</td>
<td><strong>ENCR_3DES</strong></td>
<td><strong>PRF_HMAC_SHA1</strong></td>
<td><strong>AUTH_HMAC_SHA1_96</strong></td>
</tr>
<tr>
<td>Part F</td>
<td><strong>ENCR_3DES</strong></td>
<td><strong>PRF_HMAC_SHA2_256</strong></td>
<td><strong>AUTH_HMAC_SHA1_96</strong></td>
</tr>
<tr>
<td>Part G</td>
<td><strong>ENCR_3DES</strong></td>
<td><strong>PRF_HMAC_SHA1</strong></td>
<td><strong>AUTH_HMAC_SHA2_256_128</strong></td>
</tr>
<tr>
<td>Part H</td>
<td><strong>ENCR_3DES</strong></td>
<td><strong>PRF_HMAC_SHA1</strong></td>
<td><strong>AUTH_HMAC_SHA1_96</strong></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

**Part A: Encryption Algorithm ENCR_AES_CBC (ADVANCED)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.

Part B: Encryption Algorithm ENCR_AES_CTR (ADVANCED)
This test case is deleted at revision 1.0.4.

Part C: Pseudo-Random Function PRF_AES128_CBC (ADVANCED)
9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link B.
11. TN1 responds with an IKE_SA_INIT response to the NUT.
12. Observe the messages transmitted on Link B.

Part D: Integrity Algorithm AUTH_AES_XCBC_96 (ADVANCED)
13. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link B.
15. TN1 responds with an IKE_SA_INIT response to the NUT.
16. Observe the messages transmitted on Link B.

Part E: D-H Group Group 14 (ADVANCED)
17. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link B.
19. TN1 responds with an IKE_SA_INIT response to the NUT.
20. Observe the messages transmitted on Link B.

Part F: PRF PRF_HMAC_SHA2_256 (ADVANCED)
21. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
22. Observe the messages transmitted on Link B.
23. TN1 responds with an IKE_SA_INIT response to the NUT.
24. Observe the messages transmitted on Link B.

Part G: Integrity Algorithm AUTH_HMAC_SHA2_256_128 (ADVANCED)
25. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link B.
27. TN1 responds with an IKE_SA_INIT response to the NUT.
28. Observe the messages transmitted on Link B.

Part H: D-H Group Group 24 (ADVANCED)
29. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
30. Observe the messages transmitted on Link B.
31. TN1 responds with an IKE_SA_INIT response to the NUT.
32. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_AES_CBC",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed
algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 1.

**Part B**

This test case is deleted at revision 1.0.4.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_AES128_CBC", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 9.

**Part D**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_AES_XCBC_96" and "D-H Group 2" as proposed algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 9.

**Part E**

**Step 18: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 14" as proposed algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 9.

**Part F**

**Step 22: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA2_256", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 24: Judgment #2**
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 9.

**Part G**

**Step 26: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA2_256", "AUTH_HMAC_SHA2_256_128" and "D-H Group 2" as proposed algorithms.

**Step 28: Judgment #2**
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 25.

**Part H**

**Step 30: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 24" as proposed algorithms.

**Step 32: Judgment #2**
The NUT transmits an IKE_AUTH request which is cryptographically protected by the proposed algorithms in Step 29.

**Possible Problems:**

- None.
Test IKEv2.SGW.I.1.1.6.2: Cryptographic Algorithm Negotiation for CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-Shared key.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  From part A to part G, configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>IKE_AUTH exchanges Algorithms</th>
<th>Encryption</th>
<th>Integrity</th>
<th>Extended Sequence Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR AES CBC</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR AES CTR</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR NULL</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>Part D</td>
<td>ENCR 3DES</td>
<td>AUTH_AES_XCBC 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>Part E</td>
<td>ENCR 3DES</td>
<td>NONE</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>Part F</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC SHA1 96</td>
<td>Extended Sequence Numbers</td>
</tr>
<tr>
<td>Part G</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC SHA2 256_128</td>
<td>No Extended Sequence Numbers</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1 See Common Packet #2

IPv6 FORUM TECHNICAL DOCUMENT 632 IPv6 Ready Logo Program IKEv2
Part A: Encryption Algorithm ENCR_AES_CBC (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link B.

Part B: Encryption Algorithm ENCR_AES_CTR (ADVANCED)
10. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
11. Observe the messages transmitted on Link B.
12. TN1 responds with an IKE_SA_INIT response to the NUT.
13. Observe the messages transmitted on Link B.
14. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
15. TH2 transmits an Echo Request to TH1.
16. Observe the messages transmitted on Link A.
17. TH1 transmits an Echo Reply to TH2.
18. Observe the messages transmitted on Link B.

Part C: Encryption Algorithm ENCR_NULL (ADVANCED)
19. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link B.
21. TN1 responds with an IKE_SA_INIT response to the NUT.
22. Observe the messages transmitted on Link B.
23. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
24. TH2 transmits an Echo Request to TH1.
25. Observe the messages transmitted on Link A.
26. TH1 transmits an Echo Reply to TH2.
27. Observe the messages transmitted on Link B.

Part D: Integrity Algorithm AUTH_AES_XCBC_96 (ADVANCED)
28. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
29. Observe the messages transmitted on Link B.
30. TN1 responds with an IKE_SA_INIT response to the NUT.
31. Observe the messages transmitted on Link B.
32. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
33. TH2 transmits an Echo Request to TH1.
34. Observe the messages transmitted on Link A.
35. TH1 transmits an Echo Reply to TH2.
36. Observe the messages transmitted on Link B.

Part E: Integrity Algorithm NONE (ADVANCED)
IPv6 FORUM TECHNICAL DOCUMENT

37. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
38. Observe the messages transmitted on Link B.
39. TN1 responds with an IKE_SA_INIT response to the NUT.
40. Observe the messages transmitted on Link B.
41. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
42. TH2 transmits an Echo Request to TH1.
43. Observe the messages transmitted on Link A.
44. TH1 transmits an Echo Reply to TH2.
45. Observe the messages transmitted on Link B.

Part F: Extended Sequence Numbers (ADVANCED)
46. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
47. Observe the messages transmitted on Link B.
48. TN1 responds with an IKE_SA_INIT response to the NUT.
49. Observe the messages transmitted on Link B.
50. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
51. TH2 transmits an Echo Request to TH1.
52. Observe the messages transmitted on Link A.
53. TH1 transmits an Echo Reply to TH2.
54. Observe the messages transmitted on Link B.

Part G: Integrity Algorithm AUTH_HMAC_SHA2_256_128 (ADVANCED)
55. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
56. Observe the messages transmitted on Link B.
57. TN1 responds with an IKE_SA_INIT response to the NUT.
58. Observe the messages transmitted on Link B.
59. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
60. TH2 transmits an Echo Request to TH1.
61. Observe the messages transmitted on Link A.
62. TH1 transmits an Echo Reply to TH2.
63. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_AES_CBC", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.
**Part B**

**Step 11: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 13: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_AES_CTR", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 16: Judgment #3**
The NUT forwards an Echo Request.

**Step 18: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Part C**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 22: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_NULL", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 25: Judgment #3**
The NUT forwards an Echo Request.

**Step 27: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Part D**

**Step 29: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 31: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 34: Judgment #3**
The NUT forwards an Echo Request.

**Step 36: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Part E**

**Step 38: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 40: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "NONE" and "No Extended Sequence Numbers" as proposed algorithms. However, the transform indicating "NONE" can be omitted.

Step 43: Judgment #3
The NUT forwards an Echo Request.

Step 45: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Part F

Step 47: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 49: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1" and "Extended Sequence Numbers" as proposed algorithms.

Step 52: Judgment #3
The NUT forwards an Echo Request.

Step 54: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Part G

Step 56: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 58: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA2_256_128" and "No Extended Sequence Numbers" as proposed algorithms.

Step 61: Judgment #3
The NUT forwards an Echo Request.

Step 63: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.6.3: Sending Multiple Transforms for IKE_SA

Purpose:

To verify an IKEv2 device properly transmits IKE_SA_INIT request with multiple transforms for IKE_SA.

References:

- [RFC 4306] - Sections 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration:

<table>
<thead>
<tr>
<th>Part</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td>ENCR_AES_CBC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRF_AES128_CBC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AUTH_AES_XCBC_96</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2, Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

**Part A: Multiple Encryption Algorithms (ADVANCED)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above.
2. Observe the messages transmitted on Link B.

**Part B: Multiple Pseudo-Random Functions (ADVANCED)**
3. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above.
4. Observe the messages transmitted on Link B.

**Part C: Multiple Integrity Algorithms (ADVANCED)**
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above.
6. Observe the messages transmitted on Link B.

Part D: Multiple D-H Groups (ADVANCED)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
8. Observe the messages transmitted on Link B.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Part B
Step 4: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "PRF_AES128_CBC"AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Part C
Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "AUTH_AES_XCBC_96" and "D-H Group 2" as accepted algorithms.

Part D
Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as accepted algorithms. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.6.4: Sending Multiple Proposals for IKE_SA

Purpose:

To verify an IKEv2 device properly transmits IKE_AUTH request with multiple proposals for CHILD_SA.

References:

- [RFC 4306] - Sections 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration.

<table>
<thead>
<tr>
<th>Proposal ID</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal #1</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Proposal #2</td>
<td>ENCR_AES_CBC</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_AES_XCBC_96</td>
<td>Group 14 or Group 24</td>
</tr>
</tbody>
</table>

Pre-Sequence and Cleanup Sequence
IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request with 2 SA Proposals.
SA Proposal #1 (ESP) includes "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2".
SA Proposal #2 (ESP) includes "ENCR_AES_CBC", "PRF_AES128_CBC", "AUTH_AES_XCBC_96" and "D-H Group 14". Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.
Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.6.5: Sending Multiple Transforms for CHILD_SA

Purpose:

To verify an IKEv2 device properly transmits IKE_AUTH request with multiple transforms for CHILD_SA.

References:

- [RFC 4306] - Sections 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration.

<table>
<thead>
<tr>
<th>IKE_AUTH exchanges Algorithms</th>
<th>Encryption</th>
<th>Integrity</th>
<th>ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td></td>
<td>ENCR_AES_CBC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUTH_AES_XCBC_96</td>
<td></td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ESN</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above to the TN1.
2. Observe the messages transmitted on Link B.
3. NUT transmits an IKE_AUTH request including a SA payload as described above to the TN1.
4. Observe the messages transmitted on Link B.
Part B: Multiple Integrity Algorithms (ADVANCED)
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above to the TN1.
6. Observe the messages transmitted on Link B.
7. NUT transmits an IKE_AUTH request including a SA payload as described above to the TN1.
8. Observe the messages transmitted on Link B.

Part C: Extended Sequence Numbers (ADVANCED)
9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request including a SA payload as described above to the TN1.
10. Observe the messages transmitted on Link B.
11. NUT transmits an IKE_AUTH request including a SA payload as described above to the TN1.
12. Observe the messages transmitted on Link B.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "ENCR_AES_CBC", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part B
Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 8: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part C
Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers" and "Extended Sequence Number" as proposed algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.6.6: Sending Multiple Proposals for CHILD_SA

Purpose:

To verify an IKEv2 device properly transmits IKE_AUTH request with multiple proposals for CHILD_SA.

References:

- [RFC 4306] - Sections 3.3

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the following configuration.

<table>
<thead>
<tr>
<th>Proposal ID</th>
<th>Encryption</th>
<th>Integrity</th>
<th>ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal #1</td>
<td>ESP</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
</tr>
<tr>
<td>Proposal #2</td>
<td>ESP</td>
<td>ENCR_AES_CBC</td>
<td>AUTH_AES_XCBC_96 ESN</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

Part A: (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
4. Observe the messages transmitted on Link B.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" in SA Proposal #1 (ESP) and then "ENCR_AES_CBC", "AUTH_AES_XCBC_96" and "Extended Sequence Numbers" in SA Proposal #2 (ESP) as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.6.7: Receipt of INVALID_KE_PAYLOAD

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA response with a Notify payload of type INVALID_KE_PAYLOAD.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration with enabling PFS by proposing D-H Group 2 and D-H Group 14 when rekeying. KEi payload must carry D-H Group 14 public key value in CREATE_CHILD_SA request. It is possible to use D-H Group 24 instead of D-H Group 14.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: See Common Packet #2
Packet #2: See Common Packet #6
Packet #3: See Common Packet #21
Packet #4: See Common Packet #25
Packet #5: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th></th>
<th>IPv6 Header</th>
<th>Same as Common Packet #16</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UDP Header</td>
<td>Same as Common Packet #16</td>
</tr>
<tr>
<td></td>
<td>IKEv2 Header</td>
<td>Same as Common Packet #16</td>
</tr>
<tr>
<td></td>
<td>E Payload</td>
<td>Same as Common Packet #16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>N Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>10</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>INVALID KE_PAYLOAD (17)</td>
</tr>
<tr>
<td>Notification Data</td>
<td>The accepted D-H Group # (2)</td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. Repeat Steps 6 and 7 until lifetime of SA is expired.
9. Observe the messages transmitted on Link B.
10. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response with a Notify payload of type INVALID_KE_PAYLOAD containing 2 (1024 Bit MODP) as Notification Data to the NUT.
11. Observe the messages transmitted on Link B.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 9: Judgment #4**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload must carry "D-H Group 14" public key value. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.
And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers", "D-H Group 2" and "D-H Group 14" as proposed algorithms and a Key Exchange payload which contains "D-H Group 2" public key value.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.6.8: Receipt of NO_PROPOSAL_CHOSEN

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.1.6.9: Response with inconsistent SA proposal for IKE_SA

Purpose:

To verify an IKEv2 device properly handles a response with a SA payload which is inconsistent with one of its proposals.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- **Network Topology**
  Connect the devices according to the Common Topology.
- **Configuration**
  In each part, configure the devices according to the Common Configuration.
- **Pre-Sequence and Cleanup Sequence**
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>SA Payload</td>
<td>See below</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>Ni Payload</td>
<td>Same as the Common Packet #2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Payload</th>
<th>Next Payload</th>
<th>34 (IKE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>Proposal #1</td>
<td>SA Proposal</td>
<td></td>
</tr>
<tr>
<td>Proposal Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td># of Transforms</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT. But the response includes a SA payload which has a different Transform ID from the proposed one.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT never transmits an IKE_AUTH request.

Possible Problems:

- Step 4
  The NUT may transmit or retransmit an IKE_SA_INIT request.
Test IKEv2.SGW.I.1.1.6.10: Response with inconsistent proposal for CHILD_SA

Purpose:

To verify an IKEv2 device properly handles a response with a SA payload which is inconsistent with one of its proposals.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #6</td>
</tr>
</tbody>
</table>
**IPv6 Ready Logo Program IKEv2**

<table>
<thead>
<tr>
<th>Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>SA Payload</td>
<td>See below</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
</tbody>
</table>

**Next Payload**

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>44</td>
</tr>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Proposal Length</td>
<td>40</td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of Transforms</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>See below</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>44</td>
</tr>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>0</td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>44</td>
</tr>
<tr>
<td>Next Payload</td>
<td>0 (last)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>5 (Extended Sequence Number)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>0 (No Extended Sequence Number)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>44</td>
</tr>
<tr>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>12 (AES_CBC)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Attribute</th>
<th>Attribute Type</th>
<th>Attribute Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>14 (Key Length)</td>
<td>128</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 responds with an IKE_AUTH response to the NUT. But the response includes a SA payload which has a different Transform ID from the proposed one.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT request including "ENC_APIES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT never forwards an Echo Request.

Step 9: Judgment #4
The NUT never forwards an Echo Reply with IPsec ESP using ENCR_AES_CBC and AUTH_HMAC_SHA1_96.

Possible Problems:

- Step 7
  The NUT may transmit or retransmit an IKE_AUTH request. And the NUT may notify INVALID_SPI.
Test IKEv2.SGW.I.1.1.6.11: Receipt of INVALID_KE_PAYLOAD in Initial Exchange

Purpose:

To verify an IKEv2 device properly handles IKE_SA_INIT response with a Notify payload of type INVALID_KE_PAYLOAD.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, configure the IKE_SA parameters as described as following. KEi payload must carry D-H Group 14 public key value.

<table>
<thead>
<tr>
<th>IKE_SA Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2, Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #2</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #2</td>
</tr>
<tr>
<td>N Payload</td>
<td>IKE_SA Responder’s SPI</td>
</tr>
<tr>
<td></td>
<td>See each Part</td>
</tr>
<tr>
<td>Next Payload</td>
<td>0 (No Next Payload)</td>
</tr>
</tbody>
</table>
Part A: IKE_SA Responder’s SPI is zero (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response including a Notify payload of type INVALID_KE_PAYLOAD containing 2 (1024 Bit MODP) as Notification Data to the NUT. The message’s IKE_SA Responder’s SPI is set to zero.
4. Observe the messages transmitted on Link A.

Part B: IKE_SA Responder’s SPI is not zero (ADVANCED)
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_SA_INIT response including a Notify payload of type INVALID_KE_PAYLOAD containing 2 (1024 Bit MODP) as Notification Data to the NUT. The message’s IKE_SA Responder’s SPI is set to one.
8. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload must carry "D-H Group 14" public key value. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

Step 4: Judgment #2
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. All other payloads are unchanged.

Part B
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. KEi payload must carry "D-H Group 14" public key value. Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

Step 4: Judgment #2
The NUT transmits an IKE_SA_INIT request including a Key Exchange payload which contains "D-H Group 2" public key value. All other payloads are unchanged.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.6.12: Creating an IKE_SA without a CHILD_SA

Purpose:
To verify an IKEv2 device can handle a failure of creating a CHILD_SA during the IKE_AUTH exchange.

References:
- [RFC 4718] - Sections 4.2

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

Packet #4: IKE_AUTH response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #6</td>
</tr>
<tr>
<td>IDW Payload</td>
<td>Next Payload 39 (AUTH)</td>
</tr>
<tr>
<td></td>
<td>Critical 0</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Payload Length 24</td>
</tr>
<tr>
<td>ID Type</td>
<td>IPV6_ADDR</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---</td>
</tr>
<tr>
<td>Identification Data</td>
<td>TN1’s Global Address on Link X</td>
</tr>
<tr>
<td><strong>AUTH Payload</strong></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td>Auth Method</td>
<td>2 (SK_MIC)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Authentication Data</td>
<td>any</td>
</tr>
<tr>
<td><strong>N Payload</strong></td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>Notify Message Type</td>
<td>NO_PROPOSAL_CHOSEN (14)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response with a Notify payload of type NO_PROPOSAL_CHOSEN to the NUT.
6. TN1 transmits an INFORMATIONAL request with no payloads to the NUT.
7. Observe the messages transmitted on Link A.

Observable Results:

Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits an INFORMATIONAL Response followed by an Encrypted payload with no payloads contained in it.

Possible Problems:

- None
Group 1.7. Traffic Selector Negotiation

Test IKEv2.SGW.I.1.1.7.1: Narrowing the range of members of the set of traffic selectors

Purpose:

To verify an IKEv2 device allows the responder to choose a subset of the traffic proposed by the initiator.

References:

- [RFC4306] - Section 2.9

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #5: ICMPv6 Echo Request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #21</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESP</td>
<td>Same as Common Packet #21</td>
</tr>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
</tr>
<tr>
<td></td>
<td>Other fields are same as Common Packet #21</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Same as Common Packet #21</td>
</tr>
</tbody>
</table>

Packet #6: ICMPv6 Echo Reply

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TH1’s Global Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td>TH3’s Global Address</td>
<td></td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Same as Common Packet #25</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request packet to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply packet to TH2.
9. Observe the messages transmitted on Link B.
10. TH3 transmits an Echo Request to TH1.
11. Observe the messages transmitted on Link A.
12. TH1 transmits an Echo Request to TH3.
13. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Request with IPsec ESP using corresponding algorithms.

Step 11: Judgment #5
The NUT never forwards an Echo Request.

**Step 13: Judgment #6**  
The NUT forwards an Echo Request without IPsec ESP.

**Possible Problems:**

- None.
Group 1.8. Error Handling

**Test IKEv2.SGW.I.1.1.8.1: INVALID_IKE_SPI**

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.1.8.2: INVALID_SELECTORS

This test case was deleted at revision 1.1.0.
Group 1.10 Authentication of the IKE_SA

Test IKEv2.SGW.I.1.1.10.1: Sending CERT Payload

Purpose:

To verify an IKEv2 device handles CERTREQ payload and transmits CERT payload properly.

References:

- [RFC 4306] - Sections 1.2 and 3.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
</tr>
<tr>
<td>Part B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
</tr>
<tr>
<td>Part C</td>
<td>X.509 Certificate - Signature</td>
<td>IDRFC822_ADDR</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1: See below

Packet #1: IKE_SA_INIT response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>KE Payload</td>
<td>Same as the Common Packet #2</td>
</tr>
<tr>
<td>Nr Payload</td>
<td>Next Payload 38 (CERTREQ)</td>
</tr>
</tbody>
</table>
Other fields are same as the Common Packet #2

<table>
<thead>
<tr>
<th>CERTREQ Payload</th>
<th></th>
<th>See below</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>Any</td>
<td></td>
</tr>
<tr>
<td>Certificate Encoding</td>
<td>4 (X.509 Certificate - Signature)</td>
<td></td>
</tr>
<tr>
<td>Certificate Authority</td>
<td>any</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: ID_IPV6_ADDR (ADVANCED)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT request from the NUT, TN1 responds with an IKE_SA_INIT response with a CERTREQ payload to the NUT.
4. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request. The request includes an ID payload with ID_IPV6_ADDR and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH request. The request includes an ID payload with ID_FQDN and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH request. The request includes an ID payload with ID_RFC822_ADDR and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Possible Problems:**
• None.
Test IKEv2.SGW.I.1.1.10.2: Sending CERTREQ Payload

Purpose:

To verify an IKEv2 device transmits CERTREQ payload and handles CERT payload properly.

References:

- [RFC 4306] - Sections 1.2 and 3.7

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.

- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Part</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X.509 Certificate</td>
<td>ID_IPV6_ADDR</td>
<td>TN1’s global address on Link A</td>
</tr>
<tr>
<td>B</td>
<td>X.509 Certificate</td>
<td>ID_FQDN</td>
<td>tn.example.com</td>
</tr>
<tr>
<td>C</td>
<td>X.509 Certificate</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:tn@example.com">tn@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

**Packet #1** See Common Packet #2

**Part A: ID_IPV6_ADDR (ADVANCED)**
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.

**Part B: ID_FQDN (ADVANCED)**
5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. TN1 responds with an IKE_SA_INIT response to the NUT.
8. Observe the messages transmitted on Link A.

**Part C: ID_RFC822_ADDR (ADVANCED)**

9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. TN1 responds with an IKE_SA_INIT response to the NUT.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

**Possible Problems:**

- None.
Test IKEv2.SGW.I.1.1.10.3: RSA Digital Signature

Purpose:

To verify an IKEv2 device authenticates the corresponding node by RSA Digital Signature.

References:

- [RFC 4306] - Sections 1.2 and 3.7

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Part</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>TN1’s global address on Link A</td>
</tr>
<tr>
<td>B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>tn.example.com</td>
</tr>
<tr>
<td>C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:tn@example.com">tn@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>Packet #2</th>
<th>Packet #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>See Common Packet #2</td>
<td>See below</td>
<td>See Common Packet #19</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH response
IPv6 Header | Same as Common Packet #6
---|---
UDP Header | Same as Common Packet #6
IKEv2 Header | Same as Common Packet #6
E Payload | Same as Common Packet #6
IDr Payload | Next Payload 37 (CERT)
Other fields are same as the Common Packet #6
CERT Payload | See below
AUTH Payload | Same as Common Packet #6
SA Payload | Same as Common Packet #6
TSi Payload | Same as Common Packet #6
TSr Payload | Same as Common Packet #6

<table>
<thead>
<tr>
<th>CERT Payload</th>
<th>Next Payload 39 (AUTH)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>Any</td>
</tr>
<tr>
<td>Certificate Encoding</td>
<td>4 (X.509 Certificate – Signature)</td>
</tr>
<tr>
<td>Certificate Data</td>
<td>TN1’s X.509 Certificate</td>
</tr>
</tbody>
</table>

Part A: ID_IPV6_ADDR (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response including an IDr payload as described above and a CERT payload to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.

Part B: ID_FQDN (ADVANCED)
10. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
11. Observe the messages transmitted on Link A.
12. TN1 responds with an IKE_SA_INIT response to the NUT.
13. Observe the messages transmitted on Link A.
14. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response including an IDr payload as described above and a CERT payload to the NUT.
15. TH2 transmits an Echo Request to TH1.
16. Observe the messages transmitted on Link B.
17. TH1 transmits an Echo Reply to TH2.
18. Observe the messages transmitted on Link A.

Part C: ID_RFC822_ADDR (ADVANCED)
19. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. TN1 responds with an IKE_SA_INIT response to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response including an IDr payload as described above and a CERT payload to the NUT.
24. TH2 transmits an Echo Request to TH1.
25. Observe the messages transmitted on Link B.
26. TH1 transmits an Echo Reply to TH2.
27. Observe the messages transmitted on Link A.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Part B

Step 11: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 13: Judgment #2
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Step 16: Judgment #3
The NUT forwards an Echo Request.

Step 18: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Part C

Step 20: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 22: Judgment #2
The NUT transmits an IKE_AUTH request with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Step 25: Judgment #3
The NUT forwards an Echo Request.

Step 27: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.
Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.10.4: HEX string PSK

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 2.15

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Authentication Key Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote</td>
</tr>
<tr>
<td>0xabadcafeabadcafeabadcafeabadcafe (128 bit binary string)</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>3. TN1 responds with an IKE_SA_INIT response to the NUT.</td>
</tr>
<tr>
<td>4. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

- None.
Group 1.11 Invalid values

Test IKEv2.SGW.I.1.1.11.1: Non zero RESERVED fields in IKE_SA_INIT response

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
<th>All RESERVED fields are set to one.</th>
</tr>
</thead>
</table>

Part A (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response whose RESERVED fields are set to one to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- None.
Test IKEv2.SGW.I.1.1.11.2: Non zero RESERVED fields in IKE_AUTH response

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td></td>
<td>All RESERVED fields are set to one.</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>

Part A (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH
response whose RESERVED fields are set to one to the NUT

6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.

7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Possible Problems:**

- None.
Test IKEv2.SGW.I.1.1.11.3: Version bit is set

Purpose:

To verify an IKEv2 device ignores the content of Version bit in IKE messages.

References:

- [RFC 4306] - Sections 3.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version bit is set to one.</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response whose Version bit is set to one to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.11.4: Unrecognized Notify Message Type of Error

Purpose:

To verify an IKEv2 device ignores the unrecognized Notify Message Type intended for reporting error.

References:

- [RFC 4306] - Sections 3.10.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>All fields are same as Common Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #6</td>
</tr>
</tbody>
</table>
Part A (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response with a Notify payload of unrecognized Notify Message Type value.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT never forwards an Echo Request.

Step 9: Judgment #4
The NUT never forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Possible Problems:

- None.
Test IKEv2.SGW.I.1.1.11.5: Unrecognized Notify Message Type of Status

Purpose:

To verify an IKEv2 device ignores the unrecognized Notify Message Type intended for reporting status.

References:

- [RFC 4306] - Sections 3.10.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>All fields are same as Common Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #6</td>
</tr>
<tr>
<td>Packet Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #6</td>
</tr>
<tr>
<td>E Payload</td>
<td>All fields are same as Common Packet #6</td>
</tr>
<tr>
<td>IDr Payload</td>
<td>All fields are same as Common Packet #6</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>All fields are same as Common Packet #6</td>
</tr>
<tr>
<td>SA Payload</td>
<td>All fields are same as Common Packet #6</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>All fields are same as Common Packet #6</td>
</tr>
<tr>
<td>TSr payload</td>
<td>Next Payload 41 (Notify) Other fields are same as Common Packet #6</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload 0 Critical 0 Reserved 0 Payload Length 8 Protocol ID 0 SPI Size 0 Notify Message Type 65535</td>
</tr>
</tbody>
</table>

Part A (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response with a Notify payload of unrecognized Notify Message Type value.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT never forwards an Echo Request.

Step 9: Judgment #4
The NUT never forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Possible Problems:

- None.
Group 2. The CREATE_CHILD_SA Exchange

Group 2.1. Header and Payload Formats

Test IKEv2.SGW.l.1.2.1.1: Sending CREATE_CHILD_SA request

Purpose:

To verify an IKEv2 device transmits CREATE_CHILD_SA request using properly Header and Payloads format

References:

- [RFC 4306] - Sections 1.1.2, 1.2 and 3.3.2
- [RFC 4307] - Sections 3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: IKE Header Format (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link B.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link B.

Part B: Encrypted Payload Format (BASIC)

12. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
13. Observe the messages transmitted on Link B.
14. TN1 responds with an IKE_SA_INIT response to the NUT.
15. Observe the messages transmitted on Link B.
16. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
17. TH2 transmits an Echo Request to TH1.
18. Observe the messages transmitted on Link A.
19. TH1 transmits an Echo Reply to TH2.
20. Observe the messages transmitted on Link B.
21. Repeat Steps 6 through 9 until lifetime of SA is expired.
22. Observe the messages transmitted on Link B.

Part C: Notify Payload (REKEY_SA) Format (BASIC)
23. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
24. Observe the messages transmitted on Link B.
25. TN1 responds with an IKE_SA_INIT response to the NUT.
26. Observe the messages transmitted on Link B.
27. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT
28. TH2 transmits an Echo Request to TH1.
29. Observe the messages transmitted on Link A.
30. TH1 transmits an Echo Reply to TH2.
31. Observe the messages transmitted on Link B.
32. Repeat Steps 6 through 9 until lifetime of SA is expired.
33. Observe the messages transmitted on Link B.

Part D: SA Payload Format (BASIC)
34. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
35. Observe the messages transmitted on Link B.
36. TN1 responds with an IKE_SA_INIT response to the NUT.
37. Observe the messages transmitted on Link B.
38. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT
39. TH2 transmits an Echo Request to TH1.
40. Observe the messages transmitted on Link A.
41. TH1 transmits an Echo Reply to TH2.
42. Observe the messages transmitted on Link B.
43. Repeat Steps 6 through 9 until lifetime of SA is expired.
44. Observe the messages transmitted on Link B.

Part E: Nonce Payload Format (BASIC)
45. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
46. Observe the messages transmitted on Link B.
47. TN1 responds with an IKE_SA_INIT response to the NUT.
48. Observe the messages transmitted on Link B.
49. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT
50. TH2 transmits an Echo Request to TH1.
51. Observe the messages transmitted on Link A.
52. TH1 transmits an Echo Reply to TH2.
53. Observe the messages transmitted on Link B.
54. Repeat Steps 6 through 9 until lifetime of SA is expired.
55. Observe the messages transmitted on Link B.

Part F: TSi Payload Format (BASIC)
56. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
57. Observe the messages transmitted on Link B.
58. TN1 responds with an IKE_SA_INIT response to the NUT.
59. Observe the messages transmitted on Link B.
60. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH
61. TH2 transmits an Echo Request to TH1.
62. Observe the messages transmitted on Link A.
63. TH1 transmits an Echo Reply to TH2.
64. Observe the messages transmitted on Link B.
65. Repeat Steps 6 through 9 until lifetime of SA is expired.
66. Observe the messages transmitted on Link B.

**Part G: TSr Payload Format (BASIC)**

67. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
68. Observe the messages transmitted on Link B.
69. TN1 responds with an IKE_SA_INIT response to the NUT.
70. Observe the messages transmitted on Link B.
71. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH
    response to the NUT
72. TH2 transmits an Echo Request to TH1.
73. Observe the messages transmitted on Link A.
74. TH1 transmits an Echo Reply to TH2.
75. Observe the messages transmitted on Link B.
76. Repeat Steps 6 through 9 until lifetime of SA is expired.
77. Observe the messages transmitted on Link B.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed
algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and
AUTH_HMAC_SHA1_96.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including properly formatted IKE
Header containing following values:
IPv6 FORUM TECHNICAL DOCUMENT

Figure 116 Header format

- An IKE_SA Initiator’s SPI field set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field set to Encrypted Payload (46).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to CREATE_CHILD_SA (36).
- A Flags field set to (00010000)2 = (16)10.
- A Message ID field set to the value incremented the previous IKE message’s Message ID by one.
- A Length field set to the length of the message (header + payloads) in octets.

Part B

Step 13: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 15: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 18: Judgment #3
The NUT forwards an Echo Request.

Step 20: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 22: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including properly formatted Encrypted Payload containing following values:
Part C

Step 24: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 26: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 29: Judgment #3
The NUT forwards an Echo Request.

Step 31: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 33: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including properly formatted Notify Payload containing following values:

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 12 bytes for this REKEY_SA.
- A Protocol ID field set to ESP (3).
- A SPI Size field set to the size of CHILD_SA Inbound SPI value to be rekeyed. It is 4 bytes for ESP.
- A Notify Message Type field set to REKEY_SA (16393).
- A Security Parameter Index field set to SPI value to be rekeyed.
- A Notification Data field is empty.

**Part D**

**Step 35: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 37: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 40: Judgment #3**
The NUT forwards an Echo Request.

**Step 42: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Step 44: Judgment #5**
Figure 119 SA Payload contents

The NUT transmits a CREATE_CHILD_SA request including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field set to Ni Payload (40).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.

The following proposal must be included in Proposals field.
Proposal #1
- A 0 or 2 field set to zero if this structure is the last proposal, otherwise set to 2.
- A RESREVD field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field set to 1 if this structure is the first proposal, otherwise set to 1 greater than the previous proposal.
- A Protocol ID field set to ESP (3).
- A SPI Size field set to 4.
- A # of Transforms field set to 3.
- A SPI field set to the sending entity’s SPI (4 octets value)

Transform field set to following (There are 3 Transform Structures).

Transform #1
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field set to ENCR (1).
- A RESERVED field set to zero.
- A Transform ID field set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
Part E

Step 46: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 48: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 51: Judgment #3
The NUT forwards an Echo Request.

Step 53: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 55: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including properly formatted Nonce Payload containing following values:

![Figure 123 Nonce Payload format](image_url)

- A Next Payload field set to TSi Payload (44).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Nonce Data field set to random data generated by the transmitting entity.
- The size of the Nonce must be between 16 and 256 octets.
Part F

Step 57: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 59: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 62: Judgment #3
The NUT forwards an Echo Request.

Step 64: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 66: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including properly formatted TSi Payload containing following values:

- A Next Payload field set to TSr Payload (45).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to the number of actual traffic selectors.
- A RESERVED field set to zero.

The following traffic selector must be included in Traffic Selectors field.

Figure 124 TSi Payload format
Figure 125 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to Prefix B.
- A Ending Address field set to greater than or equal to Prefix B.

Part G

Step 68: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 70: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 73: Judgment #3
The NUT forwards an Echo Request.

Step 75: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 77: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including properly formatted TSr Payload containing following values:
Figure 126 TSr Payload format

- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to 1.
- A RESERVED field set to zero.

The following traffic selector must be included in Traffic Selectors field.

Figure 127 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to Prefix Y.
- An Ending Address field set to less than or equal to Prefix Y.

Possible Problems:

- Because the destination address of Echo Request is the TN itself, TN may respond to Echo Request automatically. In that case, TH2 can send Echo Reply to TH1 instead of sending Echo Request.

- The implementation may use different SA lifetimes by the implementation policy.
that case, the tester must change the expiration time to wait CREATE_CHILD_SA request.

- CREATE_CHILD_SA request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- The implementation may not set single proposal by the implementation policy. In this case, Security Association Payload contains multiple proposals.

- Each of transforms can be located in the any order.

- The implementation may not set single traffic selector by the implementation policy. In this case, Traffic Selector Payload contains multiple proposals.
Test IKEv2.SGW.1.2.2.1: Retransmissions of CREATE_CHILD_SA requests

Purpose:

To verify an IKEv2 device retransmits CREATE_CHILD_SA request using properly Header and Payloads format

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH
   response to the NUT
6. TH1 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link A.
8. TN1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link B.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link B.
12. TN1 waits for the event of a timeout on NUT.
13. Observe the messages transmitted on Link B.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 13: Judgment #6
The NUT retransmits a CREATE_CHILD_SA request which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different lifetime of SA.
- Each NUT has the different retransmission timers.
Test IKEv2.SGW.I.1.2.2.2: Stop of retransmission of CREATE_CHILD_SA requests

Purpose:

To verify an IKEv2 device stops retransmission when it receives the corresponding response.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Request to TH2.
9. Observe the messages transmitted on Link B.

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #16</td>
</tr>
</tbody>
</table>
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link B.
12. TN1 waits for the event of a timeout on NUT.
13. Observe the messages transmitted on Link B.
14. TN1 responds with a CREATE_CHILD_SA response to the NUT.
15. TN1 waits for the event of a timeout on NUT.
16. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 13: Judgment #6
The NUT retransmits a CREATE_CHILD_SA request which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

Step 16: Judgment #7
The NUT stops the retransmissions of a CREATE_CHILD_SA request which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

Possible Problems:

- Each NUT has the different lifetime of SA.
- Each NUT has the different retransmission timers.
Group 2.3. Rekeying CHILD_SA Using a CREATE_CHILD_SA exchange

Test IKEv2.SGW.I.1.2.3.1: Close the replaced CHILD_SA

Purpose:
To verify an IKEv2 device properly handles the CREATE_CHILD_SA Exchanges to rekey CHILD_SA.

References:
- [RFC 4306] - Sections 2.8

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

Packet #1 See Common Packet #2
Packet #2 See Common Packet #6
Packet #3 See Common Packet #21
Packet #4 See Common Packet #25
Packet #5 See Common Packet #16

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. Observe the messages transmitted on Link A.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT transmits an INFORMATIONAL request with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the inbound SPI value to be deleted as SPI.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.3.2: Use of the new CHILD_SA

Purpose:

To verify an IKEv2 device properly rekeys CHILD_SA

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #6: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Source Port</td>
<td>Destination Port</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Initiator’s SPI</td>
<td>any</td>
</tr>
</tbody>
</table>

This packet is cryptographically protected by the CHILD_SA negotiated at Step 11.
**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 responds with an INFORMATIONAL response with a Delete payload to the NUT.
15. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
16. Observe the messages transmitted on Link B.
17. TH1 transmits an Echo Response to the TH2.
18. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**
Step 2: Judgment #1
The NUT transmits an IKE-SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT transmits an INFORMATIONAL request with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the inbound SPI value to be deleted as SPI.

Step 16: Judgment #7
The NUT forwards an Echo Request to the TH1.

Step 18: Judgment #8
The NUT forwards an Echo Reply with IPsec ESP using the second negotiated algorithms.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.3.3: Lifetime of CHILD_SA expires

Purpose:

To verify an IKEv2 device properly recognizes the lifetime of CHILD_SAs.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
  - In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #21</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 712 IPv6 Ready Logo Program IKEv2
Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Request to TH2.
9. Observe the messages transmitted on Link B.
10. TH2 waits for the event of a timeout on the NUT.
11. After timeout of CHILD_SA on the NUT, TH2 transmits an Echo Request to the TH1.
12. Observe the messages transmitted on Link A.
13. TH1 transmits an Echo Request to TH2.
14. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 12: Judgment #5
The NUT does not forward an Echo Request.

Step 14: Judgment #6
The NUT does not forward an Echo Reply with IPsec ESP using already expired CHILD_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.3.4: Sending Multiple Transform

Purpose:

To verify an IKEv2 device properly transmits CREATE_CHILD_SA request with multiple transforms to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration:

<table>
<thead>
<tr>
<th>CREATE_CHILD_SA exchanges Algorithms</th>
<th>Encryption</th>
<th>Integrity</th>
<th>ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A ENCR_3DES ENCR_AES_CBC</td>
<td></td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td>Part B ENCR_3DES</td>
<td></td>
<td>AUTH_HMAC_SHA1_96 AUTH_AES_XCBC</td>
<td>No ESN</td>
</tr>
<tr>
<td>Part C ENCR_3DES</td>
<td></td>
<td>AUTH_HMAC_SHA1_96 No ESN</td>
<td>ESN</td>
</tr>
</tbody>
</table>

  - Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

715

IPv6 Ready Logo Program IKEv2

Part A: Multiple Encryption Algorithms (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link B.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link B.

Part B: Multiple Integrity Algorithms (ADVANCED)

12. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
13. Observe the messages transmitted on Link B.
14. TN1 responds with an IKE_SA_INIT response to the NUT.
15. Observe the messages transmitted on Link B.
16. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
17. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP

Packet #1 See Common Packet #2
Packet #2 See Common Packet #6
Packet #3 See Common Packet #21
Packet #4 See Common Packet #25
using the first negotiated algorithms to NUT.

18. Observe the messages transmitted on Link A.
19. TH1 transmits an Echo Reply to TH2.
20. Observe the messages transmitted on Link B.
21. Repeat Steps 17 through 20 until lifetime of SA is expired.
22. Observe the messages transmitted on Link B.

**Part C: Multiple Extended Sequence Numbers (ADVANCED)**

23. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
24. Observe the messages transmitted on Link B.
25. TN1 responds with an IKE_SA_INIT response to the NUT.
26. Observe the messages transmitted on Link B.
27. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
28. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
29. Observe the messages transmitted on Link A.
30. TH1 transmits an Echo Reply to TH2.
31. Observe the messages transmitted on Link B.
32. Repeat Steps 28 through 31 until lifetime of SA is expired.
33. Observe the messages transmitted on Link B.

**Observable Results:**

*Part A*

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "ENCR_AES_CBC", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

*Part B*

**Step 13: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 15: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 18: Judgment #3**
The NUT forwards an Echo Request.

**Step 20: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

**Step 22: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

*Part C*

**Step 24: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 26: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 29: Judgment #3**
The NUT forwards an Echo Request.

**Step 31: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

**Step 33: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers" and "Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.3.5: Sending Multiple Proposal

Purpose:

To verify an IKEv2 device properly transmits CREATE_CHILD_SA request with multiple proposals to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following configuration:

<table>
<thead>
<tr>
<th>CREATE_CHILD_SA exchanges Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal ID</td>
</tr>
<tr>
<td>Part A</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

Packet #1 See Common Packet #2
Packet #2 See Common Packet #6
Packet #3 See Common Packet #21
Packet #4 See Common Packet #25

Part A: (ADVANCED)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" in SA Proposal #1 (ESP) and then "ENCR_AES_CBC", "AUTH_AES_XCBC_96" and "Extended Sequence Numbers" in SA Proposal #2 (ESP) as accepted algorithms.

Possible Problems:
- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.3.6: Rekeying Failure

Purpose:

To verify an IKEv2 device properly handles rekeying failure.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
  - In addition, set IKE_SA Lifetime to 30 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 720 IPv6 Ready Logo Program IKEv2
Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to the NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying IKE_SA from the NUT, TN1 rejects the NUT’s proposal. TN1 responds with a CREATE_CHILD_SA response with a Notify of type NO_PROPOSAL_CHOSEN.
13. TN1 transmits an INFORMATIONAL request for liveness check to the NUT.
14. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request for rekeying IKE_SA. The request includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 14: Judgment #6
The NUT never responds with an INFORMATIONAL response to an INFORMATIONAL request.
Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.3.7: Perfect Forward Secrecy

Purpose:

To verify an IKEv2 device properly rekeys CHILD_SA when Perfect Forward Secrecy enables.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.

- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds. Enable PFS.

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
### IPv6 FORUM TECHNICAL DOCUMENT

#### Packet #5: CREATE CHILD SA response

<table>
<thead>
<tr>
<th>Packet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #7</td>
<td>This packet is cryptographically protected by the CHILD SA negotiated at Step 11.</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #2</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #1</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>

### Packet #8

IPv6 Header
Same as the Common Packet #16
UDP Header
Same as the Common Packet #16
IKEv2 Header
Same as the Common Packet #16
E Payload
Same as the Common Packet #16
N Payload
Same as the Common Packet #16
N
Same as the Common Packet #16
Packet #6: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #18</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #18</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #18</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Delete Payload</th>
<th>Next Payload</th>
<th>Critical</th>
<th>Reserved</th>
<th>Payload Length</th>
<th>Protocol ID</th>
<th>SPI Size</th>
<th># of SPIs</th>
<th>Security Parameter Index(es) (SPI)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>3 (ESP)</td>
<td>4</td>
<td>1</td>
<td>SPI negotiated by Initial Exchange</td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 responds with an INFORMATIONAL response with a Delete payload to the NUT.
15. TH2 transmits an Echo Request to the TH1. TN1 forwards an Echo Request with IPsec ESP using the newly negotiated algorithms to NUT.
16. Observe the messages transmitted on Link B.
17. TH1 transmits an Echo Response to the TH2.
18. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotitated algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT transmits an INFORMATIONAL request with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the inbund SPI value to be deleted as SPI.

Step 16: Judgment #7
The NUT forwards an Echo Request to the TH1.

Step 18: Judgment #8
The NUT forwards an Echo Reply with IPsec ESP using the second negotitated algorithms.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.3.8: Use of the old CHILD_SA

Purpose:

To verify an IKEv2 device properly handles new CHILD_SA and old CHILD_SA

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- **Network Topology**
  Connect the devices according to the Common Topology.
- **Configuration**
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- **Pre-Sequence and Cleanup Sequence**
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1 | See Common Packet #2
---|---
Packet #2 | See Common Packet #6
Packet #3 | See Common Packet #21
Packet #4 | See Common Packet #25
Packet #5 | See Common Packet #16
Packet #6 | This packet is cryptographically protected by the CHILD_SA negotiated at Step 5.
Packet #7 | See Common Packet #25

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms again.
14. Observe the messages transmitted on Link B.
15. TH1 transmits an Echo Response to TH2.
16. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 14: Judgment #6**
The NUT forwards an Echo Request to the TH1.

**Step 16: Judgment #8**
The NUT forwards an Echo Reply with IPsec ESP. The NUT can use both the first CHILD_SA and the new CHILD_SA.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Group 2.4. Rekeying IKE_SAs Using a CREATE_CHILD_SA exchange

Test IKEv2.SGW.I.1.2.4.1: Close the replaced IKE_SA

Purpose:
To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:
- [RFC 4306] - Sections 2.8

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply to TH2.

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #12</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See Common Packet #18</td>
</tr>
<tr>
<td>Packet #7</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #8</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>
9. Observe the messages transmitted on Link B.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 responds with an INFORMATIONAL response to close the replaced IKE_SA.
15. TH2 transmits an Echo Request to TH1. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms inherited from the replaced IKE_SA.
16. Observe the messages transmitted on Link A.
17. TH1 transmits an Echo Reply to TH2.
18. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT transmits an INFORMATIONAL request with a Delete payload to close the replaced IKE_SA.

Step 16: Judgment #7
The NUT forwards an Echo Request.

Step 18: Judgment #8
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms inherited from the replaced IKE_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.4.2: Use of the new IKE_SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
<table>
<thead>
<tr>
<th>Packet #</th>
<th>See Common Packet #</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>#2</td>
</tr>
<tr>
<td>#2</td>
<td>#6</td>
</tr>
<tr>
<td>#3</td>
<td>#21</td>
</tr>
<tr>
<td>#4</td>
<td>#25</td>
</tr>
<tr>
<td>#5</td>
<td>#12</td>
</tr>
<tr>
<td>#6</td>
<td>#18</td>
</tr>
<tr>
<td>#7</td>
<td>#17</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link B.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 responds with an INFORMATIONAL response to an INFORMATIONAL request to close the replaced IKE_SA.
15. TN1 transmits an INFORMATIONAL request with no payloads cryptographically protected by new IKE_SA.
16. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT transmits an INFORMATIONAL request with a Delete payload to close the replaced IKE_SA.

Step 16: Judgment #7
The NUT responds with an INFORMATIONAL response with no payloads cryptographically protected by the new IKE_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.4.3: Lifetime of IKE_SA expires

Purpose:
To verify an IKEv2 device properly recognizes the lifetime of IKE_SA.

References:
- [RFC 4306] - Sections 2.8

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A: (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #1</td>
</tr>
<tr>
<td>Packet #2</td>
</tr>
<tr>
<td>Packet #3</td>
</tr>
<tr>
<td>Packet #4</td>
</tr>
</tbody>
</table>
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an INFORMATIONAL request with no payloads to the NUT.
7. Observe the messages transmitted on Link B.
8. TN1 waits for the event of a timeout on the NUT.
9. After timeout of CHILD_SA on the NUT, TN1 transmits an INFORMATIONAL request with no payloads using already expired IKE_SA.
10. Observe the messages transmitted on Link B.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT responds with an INFORMATIONAL response with no payloads.

**Step 10: Judgment #4**
The NUT does not respond with an INFORMATIONAL response with no payloads using already expired IKE_SA.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.4.4: Sending Multiple Transform

Purpose:

To verify an IKEv2 device properly transmits CREATE_CHILD_SA request with multiple transforms to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
  - In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.

<table>
<thead>
<tr>
<th>CREATE_CHILD_SA exchanges Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td>ENCR_AES_CBC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td>PRF_AES128_CBC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td></td>
<td>AUTH_AES_XCBC_96</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part D</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2, Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: Multiple Encryption Algorithms (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo-Random Functions (ADVANCED)

12. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
13. Observe the messages transmitted on Link A.
14. TN1 responds with an IKE_SA_INIT response to the NUT.
15. Observe the messages transmitted on Link A.
16. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
17. TH2 transmits an Echo Request to TH1.
18. Observe the messages transmitted on Link B.
19. TH1 transmits an Echo Reply to TH2.
20. Observe the messages transmitted on Link A.
21. Repeat Steps 17 through 20 until lifetime of SA is expired.
22. Observe the messages transmitted on Link A.

**Part C: Multiple Integrity Algorithms (ADVANCED)**
23. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
24. Observe the messages transmitted on Link A.
25. TN1 responds with an IKE_SA_INIT response to the NUT.
26. Observe the messages transmitted on Link A.
27. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
28. TH2 transmits an Echo Request to TH1.
29. Observe the messages transmitted on Link B.
30. TH1 transmits an Echo Reply to TH2.
31. Observe the messages transmitted on Link A.
32. Repeat Steps 28 through 31 until lifetime of SA is expired.
33. Observe the messages transmitted on Link A.

**Part D: Multiple D-H Groups (ADVANCED)**
34. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
35. Observe the messages transmitted on Link A.
36. TN1 responds with an IKE_SA_INIT response to the NUT.
37. Observe the messages transmitted on Link A.
38. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
39. TH2 transmits an Echo Request to TH1.
40. Observe the messages transmitted on Link B.
41. TH1 transmits an Echo Reply to TH2.
42. Observe the messages transmitted on Link A.
43. Repeat Steps 39 through 42 until lifetime of SA is expired.
44. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.
Part B

**Step 13: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 15: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 18: Judgment #3**
The NUT forwards an Echo Request.

**Step 20: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 22: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "PRF_AES128_CBC", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Part C

**Step 24: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 26: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 29: Judgment #3**
The NUT forwards an Echo Request.

**Step 31: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 33: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "AUTH_AES_XCBC_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

Part D

**Step 35: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 37: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 40: Judgment #3**
The NUT forwards an Echo Request.

**Step 42: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 44: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2" and "D-H Group 14" as proposed algorithms. Depending on configuration, it is possible to use D-H Group 24 instead of G-H group 14.
And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.4.5: Sending Multiple Proposal

Purpose:

To verify an IKEv2 device properly transmits CREATE_CHILD_SA request with multiple proposal to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.

<table>
<thead>
<tr>
<th>Proposal</th>
<th>Protocol ID</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>Proposal #1</td>
<td>IKE</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
</tr>
<tr>
<td>Proposal #2</td>
<td>IKE</td>
<td>ENCR_AES_CBC</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_AES_XCBC_96</td>
<td>Group 14 or Group 24</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: Multiple Encryption Algorithms (ADVANCED)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request with 2 SA Proposals. SA Proposal #1 (ESP) includes "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2". SA Proposal #2 (ESP) includes "ENCR_AES_CBC", "PRF_AES128_CBC", "AUTH_AES_XCBC_96" and "D-H Group 14". Depending on configuration, it is possible to use D-H Group 24 instead of D-H Group 14.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.4.6: Use of the old IKE_SA

Purpose:

To verify an IKEv2 device properly handles new IKE_SA and old IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Connect the devices according to the Common Topology.</td>
</tr>
<tr>
<td>Packet #1</td>
<td>See Common Packet #2</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #12</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See Common Packet #17 (Use old IKE_SA)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request to rekey IKE_SA from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. TN1 transmits an INFORMATIONAL request with no payload to the NUT. The message is encrypted by the old IKE_SA.
14. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 14: Judgment #6
The NUT transmits an INFORMATIONAL response with no payload. The message is encrypted by the old IKE_SA.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.4.7: Changing PRFs when rekeying the IKE_SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8
- [RFC 4718] - Sections 5.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
  Configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>IKE_SA Rekeying Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
<td>PRF_AES128_XCBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #5: CREATE_CHILD_SA response

Packet #5 is same as Common Packet #12 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform Type D-H is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>8</td>
<td>2 (PRF)</td>
<td>0</td>
<td>4 (PRF_AES128_XCBC)</td>
</tr>
</tbody>
</table>
**Part A: (ADVANCED)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link B.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 responds with an INFORMATIONAL response to an INFORMATIONAL request to close the replaced IKE_SA.
15. TN1 transmits an INFORMATIONAL request with no payloads cryptographically protected by new IKE_SA.
16. Observe the messages transmitted on Link A.

***Observable Results:***

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "PRF_AES128_XCBC", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA’s SPI value in the SPI field.

**Step 13: Judgment #6**
The NUT transmits an INFORMATIONAL request with a Delete payload to close the replaced IKE_SA.

**Step 16: Judgment #7**
The NUT responds with an INFORMATIONAL response with no payloads cryptographically protected by the new IKE_SA.
Possible Problems:

• Each NUT has the different lifetime of SA.
Group 2.5. Creating New CHILD_SAs with the CREATE_CHILD_SA Exchanges

Test IKEv2.SGW.I.1.2.5.1: Create new CHILD_SA by sending CREATE_CHILD_SA request

Purpose:

To verify an IKEv2 device properly handles the CREATE_CHILD_SA Exchanges to generate new CHILD_SAs.

References:

- [RFC 4306] - Sections 1.1.2,1.2 and 3.3.2
- [RFC 4307] - Sections 3
- [RFC 4718] - Sections 4.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See below</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #6</td>
</tr>
</tbody>
</table>
### IPv6 FORUM TECHNICAL DOCUMENT

**IPv6 Ready Logo Program IKEv2**

<table>
<thead>
<tr>
<th>Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #6</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #6</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #6</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See below</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix B:0000:0000:0000:0000:0000:0000:0000:0000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>6 (TCP)</td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix Y:0000:0000:0000:0000:0000:0000:0000:0000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. NUT starts to negotiate new CHILD_SA with TN1 by sending CREATE_CHILD_SA request.
7. Observe the messages transmitted on Link B.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Possible Problems:**

- None.
Test IKEv2.SGW.I.1.2.5.2: Receipt of cryptographically valid message on the new SA

Purpose:

To verify an IKEv2 device properly handles the CREATE_CHILD_SA Exchanges to generate new CHILD_SAs.

References:

- [RFC 4306] - Sections 1.1.2, 1.2 and 3.3.2
- [RFC 4307] - Sections 3
- [RFC 4718] - Sections 4.1

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1  See Common Packet #2
Packet #2  See below
Packet #3  See Common Packet #21
Packet #4  See Common Packet #25
Packet #5  See below
  This packet is cryptographically protected by the
  CHILD_SA negotiated at Step 1 to Step 5.
Packet #6  See below
Packet #7  See below
Packet #8  See Common Packet #21
Packet #9  See Common Packet #25
Packet #10 See below
  This packet is cryptographically protected by the
  CHILD_SA negotiated at Step 1 to Step 5.
- Packet #2: IKE_AUTH response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #4</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSi Payload Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH1’s Global Address on Link B</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH1’s Global Address on Link B</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH2’s Global Address on Link Y</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH2’s Global Address on Link Y</td>
<td></td>
</tr>
</tbody>
</table>

- Packet #5: Echo Request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td></td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>ESP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security Parameter Index</td>
<td></td>
<td>CHILD_SA’s SPI value used by this message</td>
</tr>
<tr>
<td>Sequence Number</td>
<td></td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
</tr>
<tr>
<td>Payload Data</td>
<td></td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td>Padding</td>
<td></td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td>Pad Length</td>
<td></td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td>Next Header</td>
<td>41 (IPv6)</td>
<td></td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td></td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TH3’s Global Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td></td>
<td>TH1’s Global Address</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td></td>
<td>Type 128</td>
</tr>
<tr>
<td>Code</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Identifier</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Sequence Number</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Payload Data</td>
<td>0x0000000000000000</td>
<td></td>
</tr>
</tbody>
</table>

- Packet #6: Echo Request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TH1’s Global Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td></td>
<td>TH3’s Global Address</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td></td>
<td>Type 128</td>
</tr>
<tr>
<td>Code</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Packet #7: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #4</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #4</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Traffic Selectors See below</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #4</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Traffic Selectors See below</td>
</tr>
</tbody>
</table>

TSi Payload Traffic Selector

<table>
<thead>
<tr>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH1’s Global Address on Link B</td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH1’s Global Address on Link B</td>
</tr>
</tbody>
</table>

TSr Payload Traffic Selector

<table>
<thead>
<tr>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH3’s Global Address on Link Y</td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH3’s Global Address on Link Y</td>
</tr>
</tbody>
</table>

Packet #10: Echo Request

IPv6 Header | Source Address | TN1’s Global Address on Link X
Destination Address | NUT’s Global Address on Link A
ESP | Security Parameter Index | CHILD_SA’s SPI value used by this message
Sequence Number | The value incremented the previous encrypted packet’s Sequence Number by one.
Payload Data | Subsequent data encrypted by underlying encryption algorithm
Padding | Any value which to be a multiple of the encryption block size
Pad Length | The length of the Padding field
Next Header | 41 (IPv6)
Integrity Check Value | The checksum must be valid by calculation according to the manner described in RFC.
IPv6 Header | Source Address | TH3’s Global Address
Destination Address | TH1’s Global Address
ICMPv6 Header | Type | 128
Code | 0
Identifier | any
Sequence Number | any
Payload Data | 0x0000000000000000

Packet #11: Echo Reply

IPv6 Header | Source Address | TH1’s Global Address
Destination Address | TH3’s Global Address
ICMPv6 Header | Type | 129
Code | 0
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Identifier</th>
<th>any</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence Number</td>
<td>any</td>
</tr>
<tr>
<td>Payload Data</td>
<td>0x0000000000000000</td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request packet to TH1.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply packet to TH2.
9. Observe the messages transmitted on Link B.
10. TH3 transmits an Echo Request packet to TH1.
11. Observe the messages transmitted on Link A.
12. TH1 transmits an Echo Request packet to TH3.
13. Observe the messages transmitted on Link B.
14. NUT starts to negotiate new CHILD_SA with TN1 by sending CREATE_CHILD_SA request.
15. Observe the messages transmitted on Link B.
16. After a reception of CREATE_CHILD_SA request from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT with following Traffic Selector
17. TH2 transmits an Echo Request packet to TH1.
18. Observe the messages transmitted on Link A.
19. TH1 transmits an Echo Reply packet to TH2.
20. Observe the messages transmitted on Link B.
21. TH3 transmits an Echo Request packet to TH1.
22. Observe the messages transmitted on Link A.
23. TH1 transmits an Echo Reply packet to TH3.
24. Observe the messages transmitted on Link B.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Request with IPsec ESP using corresponding algorithms.

**Step 11: Judgment #5**
The NUT never forwards an Echo Request.
Step 13: Judgment #6
The NUT never forwards an Echo Request.

Step 15: Judgment #5
The NUT transmits a CREATE_CHILD_SA request including including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 11: Judgment #5
The NUT forwards an Echo Request.

Step 13: Judgment #6
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Group 2.6. Exchange Collisions

Test IKEv2.SGW.I.1.2.6.1: Simultaneous CHILD_SA Close

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.2: Simultaneous IKE_SA Close

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.3: Simultaneous CHILD_SA Rekeying

Purpose:

To verify an IKEv2 device properly handles simultaneous CREATE_CHILD_SA Exchanges to rekey CHILD_SA.

References:

- [RFC 4718] - Sections 5.11.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
<table>
<thead>
<tr>
<th>Packet #</th>
<th>See Common Packet #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #1</td>
<td>See Common Packet #2</td>
</tr>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #15</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See Common Packet #16</td>
</tr>
<tr>
<td>Packet #7</td>
<td>See below</td>
</tr>
</tbody>
</table>
Packet #7: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request's Message ID</td>
</tr>
</tbody>
</table>

| E Payload | Next Payload | 42 (D) |
|           | Critical | 0 |
|           | Reserved | 0 |
|           | Payload Length | any |
| Initialization Vector | The same value as block length of the underlying encryption algorithm |
| Encrypted IKE Payloads | Subsequent payloads encrypted by underlying encryption algorithm |
| Padding | Any value which to be a multiple of the encryption block size |
| Pad Length | The length of the Padding field |

| D Payload | Next Payload | 0 |
|           | Critical | 0 |
|           | Reserved | 0 |
| Payload Length | 12 |
| Protocol ID | 3 (ESP) |
| SPI Size | 4 |
| # of SPIs | 1 |

Packet #8: INFORMATIONAL response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request's Message ID</td>
</tr>
</tbody>
</table>

| E Payload | Next Payload | 42 (D) |
|           | Critical | 0 |
Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 and 9 until lifetime of SA expires.
11. Observe the messages transmitted on Link A.
12. TN1 transmits a CREATE_CHILD_SA request to rekey CHILD_SA to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 responds with a CREATE_CHILD_SA response to the CREATE_CHILD_SA received at Step 9. The response message includes minimum Nonce Data.
15. Observe the messages transmitted on Link A.
16. TN1 responds with an INFORMATIONAL response to the INFORMATIONAL request received at Step 15.
17. Observe the messages transmitted on Link A.
18. TN1 responds with an INFORMATIONAL response to the INFORMATIONAL request received at Step 17.
19. TH2 transmits an Echo Request to TH1.
20. Observe the messages transmitted on Link B.
21. TH1 transmits an Echo Reply to TH2.
22. Observe the messages transmitted on Link A.

Observable Results:

Part A: (BASIC)

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request to rekey a CHILD_SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 15: Judgment #7
The NUT transmits an INFORMATIONAL request with a Delete Payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the inblund SPI value of the original CHILD_SA.

Step 18: Judgment #8
The NUT transmits an INFORMATIONAL request with a Delete Payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the inblund SPI value of the new CHILD_SA initiated by the NUT at Step 11.

Step 20: Judgment #9
The NUT forwards an Echo Request.

Step 22: Judgment #10
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.6.4: Simultaneous CHILD_SA Rekeying with retransmission

Purpose:

To verify an IKEv2 device properly handles simultaneous CREATE_CHILD_SA Exchanges to rekey CHILD_SA.

References:

- [RFC 4718] - Sections 5.11.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #15</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #7</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #8</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #9</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>
Packet #6: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1’s Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP Header</th>
<th>Source Port</th>
<th>500</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IKEv2 Header</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IKE_SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>The same value as corresponding request's Message ID</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>E Payload</th>
<th>Next Payload</th>
<th>42 (D)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D Payload</th>
<th>Next Payload</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td># of SPLs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Security Parameter Index</td>
<td>NUT’s inbound CHILD_SA SPI value of the original CHILD_SA</td>
</tr>
</tbody>
</table>

Packet #7: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #14</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #14</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #14</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>N Payload</th>
<th>Next Payload</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type</td>
<td>NO_PROPOSAL_CHOSEN (14)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 and 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. TN1 transmits a CREATE_CHILD_SA request to rekey CHILD_SA to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 transmits an INFORMATIONAL request with a Delete Payload to close the replaced CHILD_SA.
15. Observe the messages transmitted on Link A.
16. Observe the messages transmitted on Link A.
17. TN1 responds with a CREATE_CHILD_SA response with a Notify payload of type NO_PROPOSAL_CHOSEN to the retransmitted CREATE_CHILD_SA request.
18. TH2 transmits an Echo Request to TH1.
19. Observe the messages transmitted on Link B.
20. TH1 transmits an Echo Reply to TH2.
21. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request to rekey a CHILD_SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 15: Judgment #7
The NUT transmits an INFORMATIONAL response with a Delete Payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the inbund SPI value of the original CHILD_SA.

Step 16: Judgment #8
The NUT retransmits the same CREATE_CHILD_SA request as the message at Step 11. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 19: Judgment #9**
The NUT forwards an Echo Request.

**Step 21: Judgment #10**
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.6.5: Simultaneous IKE_SA Rekeying

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4718] - Sections 5.11.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #11</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See Common Packet #12</td>
</tr>
<tr>
<td>Packet #7</td>
<td>See Common Packet #18</td>
</tr>
<tr>
<td>Packet #8</td>
<td>See Common Packet #18</td>
</tr>
<tr>
<td>Packet #8</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 and 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. TN1 transmits a CREATE_CHILD_SA request to rekey IKE_SA to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 responds with a CREATE_CHILD_SA response to the CREATE_CHILD_SA request received at Step 11. The response message includes minimum Nonce Data to make the NUT send a message to close duplicated IKE_SA.
15. Observe the messages transmitted on Link A.
16. TN1 responds with an INFORMATIONAL response with no payload.
17. Observe the messages transmitted on Link A.
18. TN1 responds with an INFORMATIONAL response with no payload.
19. TN1 transmits an INFORMATIONAL request with no payload to the NUT. The message is cryptographically protected by the new IKE_SA initiated by TN1 at Step 12.
20. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request to rekey an IKE_SA. The message includes "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request has a SA payload including 1 (IKE) in the Protocol ID field, 8 in the SPI size field and new IKE_SA's SPI value in the SPI field.
Step 13: Judgment #6
The NUT responds a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the proposal in the SA payload Response has a SA payload including 1 (IKE) in the Protocol ID field, 8 in the SPI size field and new IKE_SA’s responder’s SPI value in the SPI field.

Step 15: Judgment #7
The NUT transmits an INFORMATIONAL request. The message’s IKE_SA Initiator’s SPI value is the IKE_SA Initiator’s SPI value of the original IKE_SA, and the message’s IKE_SA Responder’s SPI value is the IKE_SA Responder’s SPI value of the original IKE_SA. The message also has a Delete Payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.

Step 17: Judgment #8
The NUT transmits an INFORMATIONAL request. The message’s IKE_SA Initiator’s SPI value is the IKE_SA Initiator’s SPI value of the new IKE_SA initiated by the NUT at Step 9, and the message’s IKE_SA Responder’s SPI value is the IKE_SA Responder’s SPI value of the new IKE_SA initiated by the NUT at Step 9. The message also has a Delete Payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.

Step 20: Judgment #9
The NUT transmits an INFORMATIONAL response with no payload.

Possible Problems:
- Each NUT has the different lifetime of SA.
- Step 15 (INFORMATIONAL request to delete the original IKE_SA) can possibly switch the place with Step 17 (INFORMATIONAL request to delete the new IKE_SA).
Test IKEv2.SGW.I.1.2.6.6: Simultaneous IKE_SA Rekeying with retransmission

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4718] - Sections 5.11.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 60 seconds and set CHILD_SA Lifetime to 300 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #1: See Common Packet #2
Packet #2: See Common Packet #6
Packet #3: See Common Packet #21
Packet #4: See Common Packet #25
Packet #5: See Common Packet #11
Packet #6: See below

Packet #6: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TNT’s Global Address on Link X</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE, SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE, SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 (bit 3 of Flags)</td>
<td>any</td>
</tr>
</tbody>
</table>
**IPv6 FORUM TECHNICAL DOCUMENT**

### IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>V (bit 4 of Flags)</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>X (bits 6–7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td>Message ID</td>
<td>0</td>
</tr>
<tr>
<td>Length</td>
<td>any</td>
</tr>
</tbody>
</table>

**E Payload**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>42 (D)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
</tbody>
</table>

**D Payload**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1 (IKE_SA)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td># of SPIs</td>
<td>0</td>
</tr>
<tr>
<td>Security Parameter Index</td>
<td>none</td>
</tr>
</tbody>
</table>

### Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 and 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. TN1 transmits a CREATE_CHILD_SA request to rekey IKE_SA to the NUT.
13. Observe the messages transmitted on Link A.
14. TN1 transmits an INFORMATIONAL request to close the original IKE_SA. The message has a Delete Payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.
15. Observe the messages transmitted on Link A.
16. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**

The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.
Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Step 11: Judgment #5
The NUT transmits a CREATE_CHILD_SA request to rekey an IKE_SA. The message includes "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the CREATE_CHILD_SA request has a SA payload including 1 (IKE) in the Protocol ID field, 8 in the SPI size field and new IKE_SA’s SPI value in the SPI field.

Step 13: Judgment #6
The NUT responds a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the proposal in the SA payload Response has a SA payload including 1 (IKE) in the Protocol ID field, 8 in the SPI size field and new IKE_SA’s responder’s SPI value in the SPI field.

Step 15: Judgment #7
The NUT responds with an INFOMATIONAL response to the INFORMATIONAL request to close the original IKE_SA.

Step 16: Judgment #8
The NUT never retransmits a CREATE_CHILD_SA request transmitted at Step 11.

Possible Problems:

- Each NUT has the different lifetime of SA.
Test IKEv2.SGW.I.1.2.6.7: Rekeying a CHILD_SA while Closing a CHILD_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.8: Closing a New CHILD_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.9: Rekeying a New CHILD_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.10: Rekeying an IKE_SA with half-open CHILD_SAs

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.11: Rekeying a CHILD_SA while rekeying an IKE_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.12: Rekeying an IKE_SA with half-closed CHILD_SAs

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.13: Closing a CHILD_SA while rekeying an IKE_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.14: Closing an IKE_SA while rekeying an IKE_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.2.6.15: Rekeying an IKE_SA while Closing an IKE_SA

This test case was deleted at revision 1.1.0.
Group 2.7. Non zero RESERVED fields

Test IKEv2.SGW.I.1.2.7.1: Non zero RESERVED fields in CREATE_CHILD_SA response

Purpose:
To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:
- [RFC 4306] - Sections 2.5

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
  In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

Part A: (BASIC)

1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP using the first negotiated algorithms to NUT.
7. Observe the messages transmitted on Link B.
8. TH1 transmits an Echo Reply to TH2.
9. Observe the messages transmitted on Link A.
10. Repeat Steps 6 through 9 until lifetime of SA is expired.
11. Observe the messages transmitted on Link A.
12. After reception of CREATE_CHILD_SA request for rekeying from the NUT, TN1 responds with a CREATE_CHILD_SA response to the NUT. All RESERVED fields in the message are set to one.
13. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 7: Judgment #3**
The NUT forwards an Echo Request.

**Step 9: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using the first negotitated algorithms.

**Step 11: Judgment #5**
The NUT transmits a CREATE_CHILD_SA request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 13: Judgment #6**
The NUT transmits an INFORMATIONAL request with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the inbound SPI value to be deleted as SPI.

**Possible Problems:**

- Each NUT has the different lifetime of SA.
Group 3. The INFORMATIONAL Exchange

Group 3.1. Header and Payload Formats

Test IKEv2.SGW.I.1.3.1.1: Sending INFORMATIONAL Exchange

This test case was deleted at revision 1.1.0.
Group 3.2. Use of Retransmission Timers

Test IKEv2.SGW.I.1.3.2.1: Retransmission of INFORMATIONAL request

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.I.1.3.2.2: Stop of retransmission of INFORMATIONAL request

This test case was deleted at revision 1.1.0.
Group 3.3. Non zero RESERVED fields

Test IKEv2.SGW.I.1.3.3.1: Non zero RESERVED fields in INFORMATIONAL response

This test case was deleted at revision 1.1.0.
Group 3.4. Error Handling

Test IKEv2.SGW.I.1.3.4.1: INVALID_SPI

This test case was deleted at revision 1.1.0.
Section 2.1.2. Endpoint to Security Gateway Tunnel

Group 1. The Initial Exchanges

Group 1.1. Header and Payload Formats

Test IKEv2.SGW.I.2.1.1.1: Sending IKE_AUTH request

Purpose:

To verify an IKEv2 device transmits IKE_AUTH request using properly Header and Payloads format

References:

- [RFC 4306] - Sections 1.2, 2.15, 3.1, 3.2, 3.3, 3.5, 3.8, 3.10, 3.13 and 3.14

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

| Part A: IKE Header Format (BASIC) |
| 1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request. |
| 2. Observe the messages transmitted on Link B. |
| 3. TN1 responds with an IKE_SA_INIT response to the NUT. |
| 4. Observe the messages transmitted on Link B. |

| Part B: Encrypted Payload Format (BASIC) |
| 5. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request. |
| 6. Observe the messages transmitted on Link B. |
| 7. TN1 responds with an IKE_SA_INIT response to the NUT. |
8. Observe the messages transmitted on Link B.

Part C: IDi Payload Format (BASIC)
9. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link B.
11. TN1 responds with an IKE_SA_INIT response to the NUT.
12. Observe the messages transmitted on Link B.

Part D: AUTH Payload Format (BASIC)
13. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link B.
15. TN1 responds with an IKE_SA_INIT response to the NUT.
16. Observe the messages transmitted on Link B.

Part E: SA Payload Format (BASIC)
17. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link B.
19. TN1 responds with an IKE_SA_INIT response to the NUT.
20. Observe the messages transmitted on Link B.

Part F: TSi Payload Format (BASIC)
21. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
22. Observe the messages transmitted on Link B.
23. TN1 responds with an IKE_SA_INIT response to the NUT.
24. Observe the messages transmitted on Link B.

Part G: TSr Payload Format (BASIC)
25. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link B.
27. TN1 responds with an IKE_SA_INIT response to the NUT.
28. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted IKE Header containing following values:
Figure 128 Header format

- An IKE_SA Initiator’s SPI field set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field set to Encrypted Payload (46).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to IKE_AUTH (35).
- A Flags field set to (00010000)2 = (1610).
- A Message ID field set to 1.
- A Length field set to the length of the message (header + payloads) in octets.

Part B

Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 8: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted Encrypted Payload containing following values:

Figure 129 Encrypted payload
- A Next Payload field set to IDi Payload (35).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Checksum Data.
- An Initialization Vector field set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
- An Encrypted IKE Payloads field set to subsequent payloads encrypted by ENCR_3DES.
- A Padding field set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
- A Pad Length field set to the length of the Padding field.
- An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

Part C

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted ID Payload containing following values:

A Next Payload field set to AUTH Payload (39).
A Critical field set to zero.
A RESERVED field set to zero.
A Payload Length field set to length of the current payload. It is 24 bytes for ID_IPV6_ADDR.
An ID Type field set to ID_IPV6_ADDR (5).
A RESERVED field set to zero.
An Identification Data field set to the NUT address.

Part D

Step 14: Judgment #1

Figure 130 ID Payload format
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted AUTH Payload containing following values:

![Figure 131 AUTH Payload format](image)

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 28 bytes for PRF_HMAC_SHA1.
- An Auth Method field set to Shared Key Message Integrity Code (2).
- A RESERVED field set to zero.
- An Authentication Data field set to correct authentication value according to the manner described in RFC. It is 160 bytes length in PRF_HMAC_SHA1 case.

**Part E**

**Step 18: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH request including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field set to TSi Payload (44).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.

The following proposal must be included in Proposals field.
Proposal #1
- A 0 or 2 field set to zero if this structure is the last proposal, otherwise set to 2.
- A RESREVd field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field set to 1 if this structure is the first proposal, otherwise set to 1 greater than the previous proposal.
- A Protocol ID field set to ESP (3).
- A SPI Size field set to 4.
- A # of Transforms field set to 3.
- A SPI field set to the sending entity’s SPI (4 octets value)

Transform field set to following (There are 3 Transform Structures).
IPv6 FORUM TECHNICAL DOCUMENT

- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field set to INTEG (3).
- A RESERVED field set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
- A Transform Type field set to ESN (5).
- A RESERVED field set to zero.
- A Transform ID set to No Extended Sequence Numbers (0).

Part F

Step 22: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 24: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted TSi Payload containing following values:

![Figure 136 TSi Payload format]

- A Next Payload field set to TSr Payload (45).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to the number of actual traffic selectors.
- A RESERVED field set to zero.

The following traffic selector must be included in Traffic Selectors field.
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

Figure 137 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to Prefix B.
- A Ending Address field set to greater than or equal to Prefix B.

Part G

Step 26: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 28: Judgment #2
The NUT transmits an IKE_AUTH request including properly formatted TSr Payload containing following values:

Figure 138 TSr Payload format

- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to the number of actual traffic selectors.
- A RESERVED field set to zero.
The following traffic selector must be included in Traffic Selectors field.

![Figure 139 Traffic Selector](image)

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to TN1 address.
- An Ending Address field set to less than or equal to TN1 address.

Possible Problems:

- IKE_AUTH request has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- The implementation may not set single proposal by the implementation policy. In this case, Security Association Payload contains multiple proposals.

- The implementation may not set single traffic selector by the implementation policy. In this case, Traffic Selector Payload contains multiple proposals.
• Each of transforms can be located in the any order.
Test IKEv2.SGW.I.2.1.2.1.2: Use of CHILD_SA

Purpose:
To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:
- [RFC 4306] - Sections 1.2

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #6</td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. NUT starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. TN1 responds with an IKE_SA_INIT response to the NUT.
4. Observe the messages transmitted on Link B.
5. After reception of IKE_AUTH request from the NUT, TN1 responds with an IKE_AUTH response to the NUT.
6. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to NUT.
7. Observe the messages transmitted on Link A.
8. TH1 transmits an Echo Reply to TN1.
9. Observe the messages transmitted on Link B.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed
algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT forwards an Echo Request.

Step 9: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and
AUTH_HMAC_SHA1_96.

Possible Problems:

- Because the destination address of Echo Request is the TN itself, TN may respond to
  Echo Request automatically. In that case, TN1 can send Echo Reply to TH1 instead of
  sending Echo Request.
Section 2.2. Responder
Section 2.2.1. Security Gateway to Security Gateway Tunnel
Group 1. The Initial Exchanges
Group 1.1. Header and Payload Formats

Test IKEv2.SGW.R.1.1.1.1: Sending IKE_SA_INIT response

Purpose:

To verify an IKEv2 device transmits IKE_SA_INIT response using properly Header and Payloads format

References:

- [RFC4306] - Section 1.2, 2.10, 3.1, 3.2, 3.3, 3.4 and 3.9
- [RFC 4718] - Sections 7.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
</table>

Part A: IKE Header Format (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Part B: SA Payload Format (BASIC)

3. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
4. Observe the messages transmitted on Link A.

Part C: KE Payload Format (BASIC)

5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.

Part D: Nonce Payload Format (BASIC)

7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including properly formatted IKE Header containing following values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKE_SA Initiator’s SPI field</td>
<td>Set to IKE_SA Initiator’s SPI field value supplied in the first IKE_SA_INIT request message.</td>
</tr>
<tr>
<td>IKE_SA Responder’s SPI field</td>
<td>Set to a 64-bits value chosen by the NUT. It MUST not be zero.</td>
</tr>
<tr>
<td>Next Payload field</td>
<td>Set to SA Payload (33).</td>
</tr>
<tr>
<td>Major Version field</td>
<td>Set to 2.</td>
</tr>
<tr>
<td>Minor Version field</td>
<td>Set to zero.</td>
</tr>
<tr>
<td>Exchange Type field</td>
<td>Set to IKE_SA_INIT (34).</td>
</tr>
<tr>
<td>Flags field</td>
<td>Set to (00000100)2 = (4)10.</td>
</tr>
<tr>
<td>Message ID field</td>
<td>Set to zero.</td>
</tr>
<tr>
<td>Length field</td>
<td>Set to the length of the message (header + payloads) in octets.</td>
</tr>
</tbody>
</table>

Figure 140 Header format

- An IKE_SA Initiator’s SPI field set to IKE_SA Initiator’s SPI field value supplied in the first IKE_SA_INIT request message.
- An IKE_SA Responder’s SPI field set to a 64-bits value chosen by the NUT. It MUST not be zero.
- A Next Payload field set to SA Payload (33).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to IKE_SA_INIT (34).
- A Flags field set to (00000100)2 = (4)10.
- A Message ID field set to zero.
- A Length field set to the length of the message (header + payloads) in octets.

Part B

Step 4: Judgment #1
The NUT transmits an IKE_SA_INIT response including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field set to KE Payload (34).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Proposals field set to following.

A Proposals field set to following.
Proposal #1
- A 0 or 2 field set to zero (last).
- A RESREVD field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes. It is 40 bytes for this proposal according to Common Configuration.
- A Proposal # field set to 1.
- A Protocol ID field set to IKE (1).
- A SPI Size field set to zero.
- A # of Transforms field set to 4.

A Transform field set to following (There are 4 Transform Structures).

Transform #1
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field set to ENCR (1).
- A RESERVED field set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including
Header and Attribute. It is 8 bytes for PRF_HMAC_SHA1.
- A Transform Type field set to PRF (2).
- A RESERVED field set to zero.
- A Transform ID set to PRF_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field set to INTEG (3).
- A RESERVED field set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #4
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for 1024 MODP Group.
- A Transform Type field set to D-H (4).
- A RESERVED field set to zero.
- A Transform ID set to Group2 (2).

Part C
Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including properly formatted KE Payload containing following values:

![Figure 145 KE Payload format](image)

- A Next Payload field set to Nonce Payload (40).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 136 bytes for Group 2.
- A DH Group field set to Group2 (2).
- A RESERVED field set to zero.
- A Key Exchange Data field set to Diffie-Hellman public value. The length of the Key Exchange Data field must be equal to 1024bit.

Part D
Step 8: Judgment #4
The NUT transmits an IKE_SA_INIT response including properly formatted Nonce Payload containing following values:

- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Nonce Data field set to random data generated by the transmitting entity. The size of the Nonce must between 16 and 256 octets.

Possible Problems:

- IKE_SA_INIT response has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- Each of transforms can be located in the any order.
Test IKEv2.SGW.R.1.1.1.2: Sending IKE_AUTH response

Purpose:

To verify an IKEv2 device transmits IKE_AUTH response using properly Header and Payloads format

References:

- [RFC 4306] - Sections 1.2, 2.15, 3.1, 3.2, 3.3, 3.5, 3.8, 3.10, 3.13 and 3.14

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A: IKE Header Format (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.</td>
</tr>
<tr>
<td>4. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part B: Encrypted Payload Format (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>6. Observe the messages transmitted on Link A.</td>
</tr>
<tr>
<td>7. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.</td>
</tr>
<tr>
<td>8. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

Part C: IDr Payload Format (BASIC)
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
12. Observe the messages transmitted on Link A.

**Part D: AUTH Payload Format (BASIC)**

13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.

**Part E: SA Payload Format (BASIC)**

17. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
20. Observe the messages transmitted on Link A.

**Part F: TSi Payload Format (BASIC)**

21. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
24. Observe the messages transmitted on Link A.

**Part G: TSr Payload Format (BASIC)**

25. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
28. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted IKE Header containing following values:
Figure 147 Header format

- An IKE_SA Initiator’s SPI field set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field set to Encrypted Payload (46).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to IKE_AUTH (35).
- A Flags field set to \(00000100\)\(_2\) = \(4\)\(_{10}\).
- A Message ID field set to 1.
- A Length field set to the length of the message (header + payloads) in octets.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted Encrypted Payload containing following values:

Figure 148 Encrypted payload
• A Next Payload field set to IDr Payload (36).
• A Critical field set to zero.
• A RESERVED field set to zero.
• A Payload Length field set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Check sum Data.
• An Initialization Vector field set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
• An Encrypted IKE Payloads field set to subsequent payloads encrypted by ENCR_3DES.
• A Padding field set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
• A Pad Length field set to the length of the Padding field.
• An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted ID Payload containing following values:

![Figure 149 ID Payload format](image)

- A Next Payload field set to AUTH Payload (39).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 24 bytes for ID_IPV6_ADDR.
- An ID Type field set to ID_IPV6_ADDR (5).
- A RESERVED field set to zero.
- An Identification Data field set to the NUT address.

**Part D**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted AUTH Payload containing following values:

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 28 bytes for PRF_HMAC_SHA1.
- An Auth Method field set to Shared Key Message Integrity Code (2).
- A RESERVED field set to zero.
- An Authentication Data field set to correct authentication value according to the manner described in RFC. It is 160 bytes length in PRF_HMAC_SHA1 case.

Figure 150 AUTH Payload format

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 28 bytes for PRF_HMAC_SHA1.
- An Auth Method field set to Shared Key Message Integrity Code (2).
- A RESERVED field set to zero.
- An Authentication Data field set to correct authentication value according to the manner described in RFC. It is 160 bytes length in PRF_HMAC_SHA1 case.

Part E

Step 18: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 20: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field set to T Si Payload (44).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.

A Proposals field set to following.
Proposal #1

- A 0 or 2 field set to zero (last).
- A RESREVD field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field set to 1.
- A Protocol ID field set to ESP (3).
- A SPI Size field set to 4.
- A # of Transforms field set to 3.
- A SPI field set to the sending entity’s SPI (4 octets value)

Transform field set to following (There are 3 Transform Structures).

Transform #1

- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field set to ENCR (1).
- A RESERVED field set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2

- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including
Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.

- A Transform Type field set to INTEG (3).
- A RESERVED field set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
- A Transform Type field set to ESN (5).
- A RESERVED field set to zero.
- A Transform ID set to No Extended Sequence Numbers (0).

**Part F**

**Step 22: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 24: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted TSi Payload containing following values:

```
+-----------------+------------------+
| Next Payload    | Critical         |
+-----------------+------------------+
| TSr Payload     |                  |
+-----------------+------------------+
|                  |                  |
```

**Figure 155 TSi Payload format**

- A Next Payload field set to TSr Payload (45).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to 1.
- A RESERVED field set to zero.

Traffic Selectors field set to following.
### Figure 156 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to Prefix Y.
- A Ending Address field set to greater than or equal to Prefix Y.

### Part G

**Step 26: Judgment #1**

The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 28: Judgment #2**

The NUT transmits an IKE_AUTH response including properly formatted TSr Payload containing following values:

### Figure 157 TSr Payload format

- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to the number of actual traffic selectors.
- A RESERVED field set to zero.
Traffic Selectors field set to following.

![Figure 158 Traffic Selector](image)

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to Prefix B.
- An Ending Address field set to less than or equal to Prefix B.

Possible Problems:

- IKE_AUTH response has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

- Each of transforms can be located in the any order.
Test IKEv2.SGW.R.1.1.1.3: Use of CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key.

References:

- [RFC 4306] - Sections 1.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet</th>
<th>See Common Packet #1</th>
<th>See Common Packet #5</th>
<th>See Common Packet #21</th>
<th>See Common Packet #25</th>
</tr>
</thead>
</table>

**Part A (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT forwards an Echo Request.

**Step 8: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

**Possible Problems:**

- Because the destination address of Echo Request is the TN itself, TN may respond to Echo Request automatically. In that case, TH2 can send Echo Reply to TH1 instead of sending Echo Request.
Group 1.2. Use of Retransmission Timers

Test IKEv2.SGW.R.1.1.2.1: Receipt of retransmitted IKE_SA_INIT request

Purpose:

To verify an IKEv2 device transmits IKE_SA_INIT response, if a retransmission of the response is triggered.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4
- [RFC 4718] - Sections 2.2 and 2.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #1</td>
</tr>
<tr>
<td></td>
<td>(The Message ID is the same as Packet #1)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)

1. TN1 starts to negotiate with TN1 by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. Observe the messages transmitted on Link A.
4. TN1 retransmits the same IKE_SA_INIT request as the message transmitted in Step 1 to the
5. Observe the messages transmitted on Link A.

**Observable Results:**

*Part A*

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 3: Judgment #2**
The NUT never retransmits the same IKE_SA_INIT response as the response transmitted at Step 2.

**Step 5: Judgment #3**
The NUT transmits the same IKE_SA_INIT response as the response transmitted at Step 2.

**Possible Problems:**

- None.
Test IKEv2.SGW.R.1.1.2.2: Receipt of retransmitted IKE_AUTH request

Purpose:

To verify an IKEv2 device transmits IKE_AUTH response, if a retransmission of the response is triggered.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td></td>
<td>(The Message ID is the same as Packet #2)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. Observe the messages transmitted on Link A.
6. TN1 retransmits the same IKE_AUTH request as the request transmitted in Step 3 to the NUT.
7. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 5: Judgment #3**
The NUT never retransmits the same IKE_AUTH response as the response transmitted at Step 4.

**Step 7: Judgment #4**
The NUT transmits the same IKE_AUTH response as the response transmitted at Step 4.

**Possible Problems:**

- None.
Group 1.3. State Synchronization and Connection Timeouts

Test IKEv2.SGW.R.1.1.3.1: State Synchronization with ICMP messages

Purpose:

To verify an IKEv2 device synchronizes its state when it receives ICMP messages.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1 and TN1 forwards the Echo Request with IPsec ESP using corresponding algorithms to the NUT.
6. Observe the messages transmitted on Link A.
7. After reception of an Echo Request from the NUT, TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TR1 transmits an ICMP Destination Unreachable Message to the NUT.
10. TH2 transmits an Echo Request to TH1 and TN1 forwards the Echo Request with IPsec ESP using corresponding algorithms to the NUT.
11. Observe the messages transmitted on Link A.
12. After reception of an Echo Request from the NUT, TH1 transmits an Echo Reply to TH2.
13. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.
Step 11: Judgment #5
The NUT forwards an Echo Request.

Step 13: Judgment #6
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.3.2: State Synchronization with IKE messages

Purpose:

To verify an IKEv2 device synchronizes its state when it receives IKE messages.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>
Packet #5: cryptographically unprotected INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1's Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>NUT’s Global Address on Link X</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>41 (N)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
<td>37 (INFORMATIONAL)</td>
</tr>
<tr>
<td></td>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Notify Message Type</td>
<td>11 (INVALID_SPI)</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1 and TN1 forwards the Echo Request with IPsec ESP using corresponding algorithms to the NUT.
6. Observe the messages transmitted on Link A.
7. After reception of an Echo Request from the NUT, TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TR1 transmit a cryptographically unprotected INFORMATIONAL request with Notify payload of type INVALID_SPI to the NUT.
10. TH2 transmits an Echo Request to TH1 and TN1 forwards the Echo Request with IPsec ESP using corresponding algorithms to the NUT.
11. Observe the messages transmitted on Link A.
12. After reception of an Echo Request from the NUT, TH1 transmits an Echo Reply to TH2.
13. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT forwards an Echo Request.

**Step 8: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 11: Judgment #5**
The NUT forwards an Echo Request.

**Step 13: Judgment #6**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Possible Problems:**

- None
Test IKEv2.SGW.R.1.1.3.3: Close connections when receiving INITIAL_CONTACT

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.R.1.1.3.4: Receiving Liveness check

Purpose:

To verify an IKEv2 device checks whether the other endpoint is alive.

References:

- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an INFORMATIONAL request with no payloads.
6. Observe the messages transmitted on Link A.
Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an INFOMATIONAL Response followed by an Encrypted payload with no payloads contained in it.

Possible Problems:

- None
Test IKEv2.SGW.R.1.1.3.5: Receiving Delete Payload for IKE_SA

Purpose:

To verify an IKEv2 device transmits a Delete Payload, when IKE_SA is deleted.

References:

- [RFC 4306] - Sections 2.4 and 3.11

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #3: INFORMATIONAL request</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TN1’s Global Address on Link X</td>
<td>NUT’s Global Address on Link A</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>Destination Port</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td>46 (E)</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td>0</td>
</tr>
</tbody>
</table>

Packet #1 | See Common Packet #1
Packet #2 | See Common Packet #5
Packet #3 | See below
Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an INFORMATIONAL request with a Delete payload including 1 (IKE_SA) as Protocol ID, zero as SPI Size and no SPI value.
6. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 7: Judgment #3
The NUT transmits an INFORMATIONAL response with no payloads.

Possible Problems:

- None
Test IKEv2.SGW.R.1.1.3.6: Receiving Delete Payload for CHILD_SA

Purpose:

To verify an IKEv2 device transmits a Delete Payload, when CHILD_SAs are deleted.

References:

- [RFC 4306] - Sections 2.4 and 3.11

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

- Packet #3: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
<th>TN1’s Global Address on Link X</th>
<th>NUT’s Global Address on Link A</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
<td>Destination Port</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE SA Initiator’s SPI</td>
<td>IKE SA Responder’s SPI</td>
<td>any</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
<td></td>
<td>46 (E)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
### IPv6 FORUM TECHNICAL DOCUMENT

#### Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an INFORMATIONAL request with a Delete payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the TN1’s inbound SPI value to be deleted as SPI value.
6. Observe the messages transmitted on Link A.

#### Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 7: Judgment #3**
The NUT transmits an INFORMATIONAL response with a Delete payload including 3 (ESP) as Protocol ID, 4 as SPI Size and the NUT’s inbound SPI value to be deleted as SPI value.

#### Possible Problems:

- None

<table>
<thead>
<tr>
<th>Exchange Type</th>
<th>37 (INFORMATIONAL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X (bits 0-2 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>I (bit 3 of Flags)</td>
<td>any</td>
</tr>
<tr>
<td>V (bit 4 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>R (bit 5 of Flags)</td>
<td>0</td>
</tr>
<tr>
<td>X (bits 6-7 Flags)</td>
<td>0</td>
</tr>
<tr>
<td>Message ID</td>
<td>2</td>
</tr>
<tr>
<td>Length</td>
<td>any</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload 42 (D)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>any</td>
</tr>
<tr>
<td>Initialization Vector</td>
<td>The same value as block length of the underlying encryption algorithm</td>
</tr>
<tr>
<td>Encrypted IKE Payloads</td>
<td>Subsequent payloads encrypted by underlying encryption algorithm</td>
</tr>
<tr>
<td>Padder</td>
<td>Any value which to be a multiple of the encryption block size</td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
</tr>
<tr>
<td>Integrity Checksum Data</td>
<td>The Cryptographic checksum of the entire message</td>
</tr>
<tr>
<td>D Payload</td>
<td>Next Payload 0</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>12</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of SPIs</td>
<td>1</td>
</tr>
<tr>
<td>Security Parameter Index</td>
<td>NUT’s inbound CHILD_SA SPI value to be deleted</td>
</tr>
</tbody>
</table>
Group 1.4. Version Numbers and Forward Compatibility

Test IKEv2.SGW.R.1.1.4.1: Receipt of a higher minor version number

Purpose:

To verify an IKEv2 device drops a message with a higher minor version number and send a notification message.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1 See below

- Packet #1: IKE_SA_INIT request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the Common Packet #1</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>KE Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
</tbody>
</table>

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request with a higher minor version number.
2. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.4.2: Receipt of a higher major version number

Purpose:

To verify an IKEv2 device drops a message with a higher major version number and send a notification message.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the Common Packet #1</td>
</tr>
<tr>
<td>Major Version</td>
<td>3</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>KE Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>Ni Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
</tbody>
</table>

Packet #1: See below

Part A: BASIC
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an INFORMATIONAL response with a Notify payload of type INVALID_MAJOR_VERSION containing following values:
A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A SPI Size field set to zero.
- A Notify Message Type field set to INVALID_MAJOR_VERSION (5).
- A Notification Data field set to the highest version number it supports (2).

**Possible Problems:**

- None.
Test IKEv2.SGW.R.1.1.4.3: Unrecognized payload types and critical bit is not set

Purpose:

To verify an IKEv2 device ignores invalid payload types when the invalid type payload’s critical bit is not set.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

- Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>All fields are same as Common Packet #15 Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #15 Payload</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #15 Payload</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td>Invalid payload type value</td>
<td></td>
</tr>
</tbody>
</table>
Part A: Invalid payload type 1 (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 1 and the invalid payload’s critical flag is not set. The request includes a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

Part B: Invalid payload type 32 (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 32 and the invalid payload’s critical flag is not set. The request includes a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

Part C: Invalid payload type 49 (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 49 and the invalid payload’s critical flag is not set. The request includes a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

Part D: Invalid payload type 255 (BASIC)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. TN1 transmits a CREATE_CHILD_SA request including a payload with invalid payload type to the NUT. The E payload’s IKE Header Next Payload field is set to 255 and the
invalid payload’s critical flag is not set. The request includes a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.

24. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Part B**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 10: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 12: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Part C**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Part D**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.
Step 22: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 24: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.4.4: Unrecognized payload types and critical bit is set

Purpose:
To verify an IKEv2 device ignores invalid payload types when the invalid type payload’s critical bit is set.

References:
- [RFC 4306] - Sections 2.5

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

- Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>E Payload</td>
<td>All fields are same as Common Packet #13 Payload</td>
</tr>
<tr>
<td>N Payload</td>
<td>Next Payload</td>
</tr>
</tbody>
</table>
Other fields are same as Common Packet #13

<table>
<thead>
<tr>
<th>Invalid Payload</th>
<th>Next Payload</th>
<th>Critical</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>33 (SA)</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

SA Payload
All fields are same as Common Packet #13 Payload

Nt, Nr Payload
All fields are same as Common Packet #13 Payload

TSi Payload
All fields are same as Common Packet #13 Payload

Tsr Payload
All fields are same as Common Packet #13 Payload

**Part A: Invalid payload type 1 and Critical bit is set (BASIC)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits a CREATE_CHILD_SA request including a payload invalid payload type to the NUT. The CREATE_CHILD_SA request’s IKE Header Next Payload field is set to 1 and the pointed payload’s Critical bit is set.
6. Observe the messages transmitted on Link A.

**Part B: Invalid payload type 32 and Critical bit is set (BASIC)**
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN1 transmits a CREATE_CHILD_SA request including a payload invalid payload type to the NUT. The CREATE_CHILD_SA request’s IKE Header Next Payload field is set to 32 and the pointed payload’s Critical bit is set.
12. Observe the messages transmitted on Link A.

**Part C: Invalid payload type 49 and Critical bit is set (BASIC)**
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. After reception of IKE_AUTH response from the NUT, TN1 transmits a CREATE_CHILD_SA request including a payload invalid payload type to the NUT. The CREATE_CHILD_SA request’s IKE Header Next Payload field is set to 49 and the pointed payload’s Critical bit is set.
18. Observe the messages transmitted on Link A.

**Part D: Invalid payload type 255 and Critical bit is set (BASIC)**
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH response from the NUT, TN1 transmits a CREATE_CHILD_SA request including a payload invalid payload type to the NUT. The...
CREATE_CHILD_SA request’s IKE Header Next Payload field is set to 255 and the pointed payload’s Critical bit is set.

24. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response. The response has a Notify payload of type UNSUPPORTED_CRITICAL_PAYLOAD with the invalid payload type value (1).

**Part B**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response. The response has a Notify payload of type UNSUPPORTED_CRITICAL_PAYLOAD with the invalid payload type value (32).

**Part C**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response. The response has a Notify payload of type UNSUPPORTED_CRITICAL_PAYLOAD with the invalid payload type value (49).

**Part D**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.
Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response. The response has a Notify payload of type UNSUPPORTED_CRITICAL_PAYLOAD with the invalid payload type value (255).

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.4.5: Invalid Order Payloads

Purpose:

To verify an IKEv2 device properly handles IKE message with invalid order payloads.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part A (BASIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.</td>
</tr>
<tr>
<td>2. Observe the messages transmitted on Link A.</td>
</tr>
</tbody>
</table>

Observable Results:

Part A

Step 2: Judgment #1

The NUT never transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:

- None.
Group 1.5. Cookies

Test IKEv2.SGW.R.1.1.5.1: Cookies

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.R.1.1.5.2: Invalid Cookies

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.R.1.1.5.3: Interaction of COOKIE and INVALID KE PAYLOAD

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.R.1.1.5.4: Interaction of COOKIE and INVALID Ke_PAYLOAD with unoptimized Initiator

This test case was deleted at revision 1.1.0.
Group 1.6. Cryptographic Algorithm Negotiation

Test IKEv2.SGW.R.1.1.6.1: Cryptographic Algorithm Negotiation for IKE_SA

Purpose:
To verify an IKEv2 device properly handles the Initial Exchanges using Pre-Shared key.

References:
- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  From part A to part H, configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>IKE_SA_INIT exchanges Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A ENCR_AES_CBC</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part B ENCR_3DES</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part C ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 14</td>
<td></td>
</tr>
<tr>
<td>Part D ENCR_3DES</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part E ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part F ENCR_3DES</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part G ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>Part H ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

**Packet #1**: See below
Packet #1: IKE_SA_INIT request
Packet #1 is same as Common Packet #1 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform Type ENCR is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>12 (AES_CBC)</td>
<td></td>
</tr>
<tr>
<td>SA Attribute</td>
<td>Attribute Type</td>
<td>14 (Key Length)</td>
<td>Attribute Value</td>
<td>128</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Part B:
This test case is deleted at revision 1.0.4.

Part C:
SA Transform of Tranform Type PRF is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>2 (PRF)</td>
<td>0</td>
<td>4 (AES_128_XCBC)</td>
<td></td>
</tr>
</tbody>
</table>

Part D:
SA Transform of Tranform Type INTEG is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>5 (AES_XCBC_96)</td>
<td></td>
</tr>
</tbody>
</table>

Part E:
SA Transform of Tranform Type D-H is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>8</td>
<td>4 (D-H)</td>
<td>0</td>
<td>14 (2048 MODP Group)</td>
<td></td>
</tr>
</tbody>
</table>

Part F:
SA Transform of Tranform Type PRF is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>2 (PRF)</td>
<td>0</td>
<td>5 (HMAC_SHA2_256)</td>
<td></td>
</tr>
</tbody>
</table>

Part G:
SA Transform of Tranform Type INTEG is replaced by the following SA Transfrom.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>3 (INTEG)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Part H:
SA Transform of Transform Type D-H is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Reserved</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>8</td>
<td>4 (D-H)</td>
<td>0</td>
<td>24 (2048-bit MODP Group with 256-bit Prime Order Subgroup)</td>
<td></td>
</tr>
</tbody>
</table>

Part A: Encryption Algorithm ENCR_AES_CBC (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
4. Observe the messages transmitted on Link A.

Part B: Encryption Algorithm ENCR_AES_CTR (ADVANCED)
This test case was deleted at revision 1.1.0.

Part C: PRF PRF_AES128_CBC (ADVANCED)
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
12. Observe the messages transmitted on Link A.

Part D: Integrity Algorithm AUTH_AES_XCBC_96 (ADVANCED)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
16. Observe the messages transmitted on Link A.

Part E: D-H Group Group 14 (ADVANCED)
17. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
18. Observe the messages transmitted on Link A.
19. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
20. Observe the messages transmitted on Link A.

Part F: PRF PRF_HMAC_SHA2_256 (ADVANCED)
21. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
24. Observe the messages transmitted on Link A.
Part G: Integrity Algorithm AUTH_HMAC_SHA2_256_128 (ADVANCED)
25. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
26. Observe the messages transmitted on Link A.
27. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
28. Observe the messages transmitted on Link A.

Part H: D-H Group Group 24 (ADVANCED)
29. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
30. Observe the messages transmitted on Link A.
31. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request protected with the accepted proposal to the NUT.
32. Observe the messages transmitted on Link A.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_AES_CBC", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Part B
This test case was deleted at revision 1.1.0.

Part C

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_AES128_CBC", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Part D

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as accepted algorithms.

Part E

Step 18: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 14" as accepted algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part F**

**Step 22: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA2_256", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 24: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part G**

**Step 26: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA2_256_128" and "D-H Group 2" as accepted algorithms.

**Step 28: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part H**

**Step 30: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 24" as accepted algorithms.

**Step 32: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Possible Problems:**

- None.
Test IKEv2.SGW.R.1.1.6.2: Cryptographic Algorithm Negotiation for CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-Shared key.

References:

- [RFC 4306] - Sections 2.7 and 3.3

Test Setup:

- Network Topology
  
  Connect the devices according to the Common Topology.

- Configuration
  
  In each part, configure the devices according to the Common Configuration.

- Pre-Sequence and Cleanup Sequence
  
  IKEv2 on the NUT is disabled after each part.

From part A to part G, TN1 transmits an IKE_AUTH request including a SA payload which contains the transforms as follows:

<table>
<thead>
<tr>
<th>Part</th>
<th>Encryption</th>
<th>Integrity</th>
<th>Extended Sequence Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ENCR AES CBC</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>B</td>
<td>ENCR AES CTR</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>C</td>
<td>ENCR NULL</td>
<td>AUTH HMAC SHA1 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>D</td>
<td>ENCR 3DES</td>
<td>AUTH AES XCBC 96</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>E</td>
<td>ENCR 3DES</td>
<td>NONE</td>
<td>No Extended Sequence Numbers</td>
</tr>
<tr>
<td>F</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC SHA1 96</td>
<td>Extended Sequence Numbers</td>
</tr>
<tr>
<td>G</td>
<td>ENCR 3DES</td>
<td>AUTH HMAC SHA2 256_128</td>
<td>No Extended Sequence Numbers</td>
</tr>
</tbody>
</table>

Procedure:
Packet #2: IKE_AUTH request
Packet #2 is same as Common Packet #5 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform Type ENCR is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>SA Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>12 (AES_CBC)</td>
<td>14 (Key Length)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>128</td>
</tr>
</tbody>
</table>

Part B:
SA Transform of Tranform Type ENCR is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
<th>SA Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>13 (AES_CTR)</td>
<td>14 (Key Length)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>128</td>
</tr>
</tbody>
</table>

Part C:
SA Transform of Tranform Type ENCR is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>11 (ENCR_NULL)</td>
</tr>
</tbody>
</table>

Part D:
SA Transform of Tranform Type INTEG is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>5 (AES_XCBC_96)</td>
</tr>
</tbody>
</table>

Part E:
SA Transform of Tranform Type INTEG is replaced by the following SA Transform.
Part F:

SA Transform of Tranform Type ESN is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>8</td>
<td>5 (ESN)</td>
<td>0</td>
<td>1 (Extended sequence numbers)</td>
</tr>
</tbody>
</table>

Part G:

SA Transform of Tranform Type INTEG is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>12 (HMAC_SHA2_256_128)</td>
</tr>
</tbody>
</table>

Part A: Encryption Algorithm ENCR_AES_CBC (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.

Part B: Encryption Algorithm ENCR_AES_CTR (ADVANCED)
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
12. Observe the messages transmitted on Link A.
13. TH2 transmits an Echo Request to TH1.
14. Observe the messages transmitted on Link B.
15. TH1 transmits an Echo Reply to TH2.
16. Observe the messages transmitted on Link A.

Part C: Encryption Algorithm ENCR_NULL (ADVANCED)
17. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
20. Observe the messages transmitted on Link A.
21. TH2 transmits an Echo Request to TH1.
22. Observe the messages transmitted on Link B.
23. TH1 transmits an Echo Reply to TH2.
24. Observe the messages transmitted on Link A.

Part D: Integrity Algorithm AUTH_AES_XCBC_96 (ADVANCED)
25. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
28. Observe the messages transmitted on Link A.
29. TH2 transmits an Echo Request to TH1.
30. Observe the messages transmitted on Link B.
31. TH1 transmits an Echo Reply to TH2.
32. Observe the messages transmitted on Link A.

**Part E: Integrity Algorithm NONE (ADVANCED)**
33. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
34. Observe the messages transmitted on Link A.
35. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
36. Observe the messages transmitted on Link A.
37. TH2 transmits an Echo Request to TH1.
38. Observe the messages transmitted on Link B.
39. TH1 transmits an Echo Reply to TH2.
40. Observe the messages transmitted on Link A.

**Part F: Extended Sequence Numbers (ADVANCED)**
41. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
42. Observe the messages transmitted on Link A.
43. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
44. Observe the messages transmitted on Link A.
45. TH2 transmits an Echo Request to TH1.
46. Observe the messages transmitted on Link B.
47. TH1 transmits an Echo Reply to TH2.
48. Observe the messages transmitted on Link A.

**Part G: Integrity Algorithm AUTH_HMAC_SHA2_256_128 (ADVANCED)**
49. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
50. Observe the messages transmitted on Link A.
51. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request as described above to the NUT.
52. Observe the messages transmitted on Link A.
53. TH2 transmits an Echo Request to TH1.
54. Observe the messages transmitted on Link B.
55. TH1 transmits an Echo Reply to TH2.
56. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_AES_CBC", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT forwards an Echo Request.

**Step 8: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Part B**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_AES_CTR", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 14: Judgment #3**
The NUT forwards an Echo Request.

**Step 16: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Part C**

**Step 18: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_NULL", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 22: Judgment #3**
The NUT forwards an Echo Request.

**Step 24: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Part D**

**Step 26: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 28: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_AES_XCBC_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 30: Judgment #3**
The NUT forwards an Echo Request.
Step 32: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Part E

Step 34: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 36: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "NONE" and "No Extended Sequence Numbers" as accepted algorithms. However, the transform indicating "NONE" can be omitted.

Step 38: Judgment #3
The NUT forwards an Echo Request.

Step 40: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Part F

Step 42: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 44: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "Extended Sequence Numbers" as accepted algorithms.

Step 46: Judgment #3
The NUT forwards an Echo Request.

Step 48: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Part G

Step 50: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 52: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA2_256_128" and "No Extended Sequence Numbers" as accepted algorithms.

Step 54: Judgment #3
The NUT forwards an Echo Request.

Step 56: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.
Possible Problems:

- None.
# Test IKEv2.SGW.R.1.1.6.3: Receiving Multiple Transforms for IKE_SA

**Purpose:**

To verify an IKEv2 device properly handles IKE_SA_INIT request with a multiple transforms payload.

**References:**

- [RFC 4306] - Sections 3.3

**Test Setup:**

- **Network Topology**
  
  Connect the devices according to the Common Topology.

- **Configuration**
  
  In each part, configure the devices according to the Common Configuration.

- **Pre-Sequence and Cleanup Sequence**
  
  IKEv2 on the NUT is disabled after each part.

**Procedure:**

**IKE_SA_INIT exchanges Algorithms**

<table>
<thead>
<tr>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A ENCR_AES_CBC ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part B ENCR_3DES</td>
<td>PRF_AES128_CBC PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part C ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_AES_XCBC_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>Part D ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 14 or Group 24, Group 2</td>
</tr>
</tbody>
</table>

- **Packet #1 IKE_SA_INIT request**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See SA Table below</td>
</tr>
</tbody>
</table>

---

**Packet #1 See below**

From part A to part D, TN1 transmits an IKE_SA_INIT request including a SA payload which contains the transforms as follows:
Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
2. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo-Random Functions (BASIC)
3. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
4. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithms (BASIC)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
6. Observe the messages transmitted on Link A.

Part D: Multiple D-H Groups (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
8. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Part B**

**Step 4: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Part C**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Part D**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Possible Problems:**

- None.
Test IKEv2.SGW.R.1.1.6.4: Receiving Multiple Proposals for IKE_SA

Purpose:

To verify an IKEv2 device properly handles IKE_SA_INIT request with multiple proposals.

References:

- [RFC 4306] - Sections 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1 See below

From part A to part D, TN1 transmits an IKE_SA_INIT request including a SA payload which contains the proposals as follows:

<table>
<thead>
<tr>
<th>IKE_SA_INIT exchanges Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposals</td>
</tr>
</tbody>
</table>
| Part A
  Proposal #1 | IKE | ENCR_AES_CBC | PRF_HMAC_SHA1 | AUTH_HMAC_SHA1_96 | Group 2 |
  Proposal #2 | IKE | ENCR_3DES | PRF_HMAC_SHA1 | AUTH_HMAC_SHA1_96 | Group 2 |
| Part B
  Proposal #1 | IKE | ENCR_3DES | PRF_AES128_CBC | AUTH_HMAC_SHA1_96 | Group 2 |
  Proposal #2 | IKE | ENCR 3DES | PRF_HMAC_SHA1 | AUTH_HMAC_SHA1_96 | Group 2 |
| Part C
  Proposal #1 | IKE | ENCR_3DES | PRF_HMAC_SHA1 | AUTH_AES_XCBC_96 | Group 2 |
  Proposal #2 | IKE | ENCR 3DES | PRF_HMAC_SHA1 | AUTH_HMAC_SHA1_96 | Group 2 |
| Part D
  Proposal #1 | IKE | ENCR_3DES | PRF_HMAC_SHA1 | AUTH_HMAC_SHA1_96 | Group 14 or Group 24 |
  Proposal #2 | IKE | ENCR 3DES | PRF_HMAC_SHA1 | AUTH_HMAC_SHA1_96 | Group 2 |

- Packet #1 IKE_SA_INIT request

IPv6 Header | Same as the Common Packet #1
UDP Header | Same as the Common Packet #1
IKEv2 Header | Same as the Common Packet #1
SA Payload | Other fields are same as the common packet #1
### Proposal #1

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>Proposal Length</th>
<th>Proposal #</th>
<th>Protocol ID</th>
<th>SPI Size</th>
<th># of Transforms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 (more)</td>
<td>44</td>
<td>1</td>
<td>1 (IKE)</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

### Proposal #2

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>Proposal Length</th>
<th>Proposal #</th>
<th>Protocol ID</th>
<th>SPI Size</th>
<th># of Transforms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>44</td>
<td>2</td>
<td>1 (IKE)</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>8</td>
<td>4 (D-H)</td>
<td>0</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>8</td>
<td>2 (PRF)</td>
<td>0</td>
<td>3 (DES)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>8</td>
<td>1 (ENCR)</td>
<td>0</td>
<td>2 (HMAC_SHA1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>8</td>
<td>3 (INTEG)</td>
<td>0</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

| SA Transform | Reserved | | | | |
|--------------|----------|| | | |
Transform Length 8
Transform Type 4 (D-H)
Reserved 0
Transform ID 2 (1024 MODP Group)

Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
2. Observe the messages transmitted on Link A.

Part B: Multiple Pseudo-Random Functions (BASIC)
3. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
4. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithms (BASIC)
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
6. Observe the messages transmitted on Link A.

Part D: Multiple D-H Groups (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload as described above.
8. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Part B
Step 4: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Part C
Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Part D
Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:
- None
Test IKEv2 SGW.R.1.1.6.5: Receiving Multiple Transforms for CHILD_SA

Purpose:

To verify an IKEv2 device properly handles IKE_SA_INIT request with an unacceptable SA payload.

References:

- [RFC 4306] - Sections 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

From part A to part D, TN1 transmits an IKE_AUTH request including a SA payload which contains the transforms as follows:

<table>
<thead>
<tr>
<th>IKE_AUTH exchanges Algorithms</th>
<th>Encryption</th>
<th>Integrity</th>
<th>ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>ENCR_AES_CBC ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES</td>
<td>AUTH_AES_XCBC_96 AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
</tr>
<tr>
<td>Part C</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>ESN No ESN</td>
</tr>
</tbody>
</table>

Packet #1: See Common Packet #1
Packet #2: See below

- Packet #2: IKE_AUTH request

IPv6 Header Same as the Common Packet #5
<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Proposal Length</td>
</tr>
<tr>
<td></td>
<td>Proposal #</td>
</tr>
<tr>
<td></td>
<td>Proposal ID</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
</tr>
<tr>
<td></td>
<td># of Transforms</td>
</tr>
<tr>
<td></td>
<td>SPI</td>
</tr>
<tr>
<td><strong>SA Transform</strong></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
</tr>
<tr>
<td><strong>SA Transform</strong></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
</tr>
<tr>
<td><strong>SA Transform</strong></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
</tr>
<tr>
<td><strong>SA Transform</strong></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
</tr>
</tbody>
</table>

**Part A: Multiple Encryption Algorithms (BASIC)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
4. Observe the messages transmitted on Link A.

**Part B: Multiple Integrity Algorithms (BASIC)**
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
8. Observe the messages transmitted on Link A.

**Part C: Multiple Extended Sequence Numbers (BASIC)**
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.6.6: Receiving Multiple Proposals for CHILD_SA

Purpose:
To verify an IKEv2 device properly handles CHILD_SA request with an unacceptable SA payload.

References:
- [RFC 4306] - Sections 3.3

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>IKE_AUTH exchanges Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal</td>
</tr>
<tr>
<td>Part A</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Part B</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Part C</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Packet #1  See Common Packet #1
Packet #2   See below

TN1 transmits an IKE_AUTH request including a SA payload which contains the two proposals as follows:

- Packet #2: IKE_AUTH request
<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #5</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are Same as the Common Packet #5</td>
</tr>
<tr>
<td></td>
<td>SA Proposals See below</td>
</tr>
<tr>
<td>TSI Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
</tbody>
</table>

### Proposal #1

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Proposal Length</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Proposal #</td>
<td>1</td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
<td># of Transforms</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>SPI</td>
<td>Any</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0 (last)</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td>Transform Type</td>
<td>According to above configuration</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

### Proposal #2

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Proposal Length</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Proposal #</td>
<td>2</td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
<td>4</td>
<td># of Transforms</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>SPI</td>
<td>Any</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td>Transform ID</td>
<td>3 (3DES)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td>Transform ID</td>
<td>2 (HMAC_SHA1.96)</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0 (last)</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Transform Type: 5 (ESN)
Reserved: 0
Transform ID: 0 (No ESN)

**Part A: Multiple Encryption Algorithms (BASIC)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
4. Observe the messages transmitted on Link A.

**Part B: Multiple Integrity Algorithms (BASIC)**
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
8. Observe the messages transmitted on Link A.

**Part C: Multiple Extended Sequence Numbers (BASIC)**
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request including a SA payload as described above to the NUT.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including a SA Proposal with "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH response including a SA Proposal with "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH response including a SA Proposal with "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.6.7: Sending INVALID KE PAYLOAD

Purpose:

To verify an IKEv2 device properly handles a KE payload which has different D-H Group # from accepted D-H Group #.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. Enable PFS.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See below</td>
</tr>
</tbody>
</table>
Packet #3: CREATE_CHILD_SA request for rekeying CHILD_SA

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See SA Table below</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>Next Payload</td>
<td>34 (KE)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>KEi Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>264</td>
</tr>
<tr>
<td>DH Group #</td>
<td>14</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Key Exchange Data</td>
<td>DH#14 public key value</td>
</tr>
</tbody>
</table>

| TSi Payload   | Same as the Common Packet #15                   |
| T Sr Payload  | Same as the Common Packet #15                   |

**SA Payloads**

<table>
<thead>
<tr>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Proposal Length</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>Proposal #</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1 (IKE)</td>
<td></td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td># of Transforms</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>3 (3DES)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>2 (PRF)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC SHA1)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC SHA1 96)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>4 (D-H)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (1024 MODP Group)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Transform Type</td>
<td>4 (D-H)</td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Transform ID</td>
<td>14 (2048 MODP Group)</td>
<td></td>
</tr>
</tbody>
</table>

Packet #4: CREATE_CHILD_SA request for rekeying CHILD_SA

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>Payload</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>N Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as Packet #3</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>Other fields are same as the Packet #3</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>DH Group # 2</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>KE Group Data DH#2 public key value</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs. The CREATE_CHILD_SA contains a D-H Group transform to use D-H Group 2 and D-H Group 14, and a Key Exchange payload which contains 14 (D-H Group 14) as DH Group # field and the Key Exchange Data. It is possible to use D-H Group 24 instead of D-H Group 14.
6. Observe the messages transmitted on Link A.
7. After reception of CREATE_CHILD_SA response indicating INVALID_KE_PAYLOAD from the NUT, TN1 retransmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs. The CREATE_CHILD_SA contains a D-H Group transform to use D-H Group 2 and D-H Group 14, and a Key Exchange payload which contains 2 (D-H Group 2) as DH Group # field and the Key Exchange Data. It is possible to use D-H Group 24 instead of D-H Group 14.
8. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including a Notify payload of type INVALID_KE_PAYLOAD which contains 2 (D-H Group 2) as Notification Data.

**Step 8: Judgment #4**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96", "No Extended Sequence Numbers" and "D-H Group 2" as proposed algorithms.

**Possible Problems:**

- None.
Test IKEv2.SGW.R.1.1.6.8: Sending INVALID_KE_PAYLOAD in Initial Exchange

Purpose:

To verify an IKEv2 device properly handles KE payload which has different D-H Group # from accepted D-H Group #.

References:

- [RFC 4306] - Sections 2.7, 3.4 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See below</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common packet #1</td>
</tr>
</tbody>
</table>

Packet #1: IKE_SA_INIT request

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #1</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>Other fields are same as the common packet #1</td>
</tr>
<tr>
<td></td>
<td>SA Proposals See SA Table below</td>
</tr>
<tr>
<td></td>
<td>KEi Payload</td>
</tr>
<tr>
<td></td>
<td>DH Group # 14</td>
</tr>
<tr>
<td></td>
<td>Key Exchange Data DH#14 public key value</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #1</td>
</tr>
</tbody>
</table>

SA Payloads
### IPv6 FORUM TECHNICAL DOCUMENT

#### Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request including a SA payload which contains a D-H Group transform proposes using D-H Group 2 and D-H Group 14, and a Key Exchange payload which contains 14 (D-H Group 14) as DH Group # field and the Key Exchange Data. It is possible to use D-H Group 24 instead of D-H Group 14.

2. Observe the messages transmitted on Link A.

3. TN1 transmits an IKE_SA_INIT request to the NUT.

4. Observe the messages transmitted on Link A.

#### Observable Results:

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT response including a Notify payload of type INVALID_KE_PAYLOAD which contains 2 (D-H Group 2) as Notification Data. The message’s IKE_SA Responder’s SPI value is set to zero.

**Step 4: Judgment #2**

The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

### Possible Problems:

---

---
• None.
Test IKEv2.SGW.R.1.1.6.9: Creating an IKE_SA without a CHILD_SA

Purpose:
To verify that an IKEv2 device can handle a failure of creating a CHILD_SA during the IKE_AUTH exchange.

References:
- [RFC 4718] - Sections 4.2

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #2: IKE_AUTH request
Packet #2 is same as Common Packet #5 except SA Transform proposed in each test.

Part A:
SA Transform of Tranform Type ENCR is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH request with unacceptable SA proposal for the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an INFORMATIONAL request with no payloads.
6. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including a Notify type of NO_PROPOSAL_CHOSEN.

Step 6: Judgment #3
The NUT transmits an INFORMATIONAL Response followed by an Encrypted payload with no payloads contained in it.

Possible Problems:
- None
Group 1.7. Traffic Selector Negotiation

Test IKEv2.SGW.R.1.1.7.1: Narrowing Traffic Selectors

Purpose:

To verify an IKEv2 device allows the responder to choose a subset of the traffic proposed by the initiator.

References:

- [RFC4306] - Section 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration except Traffic Selector. Traffic Selector should be configured as following.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Address Range</td>
<td>Next Layer Protocol</td>
</tr>
<tr>
<td>Inbound</td>
<td>TH2</td>
<td>ANY</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT</td>
<td>ANY</td>
</tr>
</tbody>
</table>

The other packets are allowed to BYPASS IPsec protection.

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A (BASIC)

1. TN1 sends an IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 sends an IKE_SA_INIT request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request packet to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply packet to TH2.
8. Observe the messages transmitted on Link A.
9. TH3 transmits an Echo Request to TH1.
10. Observe the messages transmitted on Link B.
11. TH1 transmits an Echo Request to TH3.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. The Traffic Selector is narrowed to allow only address range of TH2.

**Step 6: Judgment #3**
The NUT forwards an Echo Request.

**Step 8: Judgment #4**
The NUT forwards an Echo Request with IPsec ESP using corresponding algorithms.

**Step 10: Judgment #5**
The NUT never forwards an Echo Request.

**Step 12: Judgment #6**
The NUT forwards an Echo Request without IPsec ESP.

**Possible Problems:**

- Because the destination address of Echo Request is the TN itself, TN may respond to Echo Request automatically. In that case, TH2 can send Echo Reply to TH1 instead of sending Echo Request.
Test IKEv2.SGW.R.1.1.7.2: TS_UNACCEPTABLE

Purpose:

To verify an IKEv2 device properly handles the Traffic Selector.

References:

- [RFC 4306] - Sections 3.10.1

Test Setup:

- **Network Topology**
  
  Connect the devices according to the Common Topology.

- **Configuration**
  
  In each part, configure the devices according to the Common Configuration except Traffic Selector. Traffic Selector should be configured as following.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Address</td>
<td>Next Layer</td>
</tr>
<tr>
<td></td>
<td>Range</td>
<td>Protocol</td>
</tr>
<tr>
<td>Inbound</td>
<td>TH2</td>
<td>ANY</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT</td>
<td>ANY</td>
</tr>
</tbody>
</table>

The other packets are allowed to BYPASS IPsec protection.

- **Pre-Sequence and Cleanup Sequence**
  
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>
Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #5</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #5</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #5</td>
</tr>
</tbody>
</table>

**TSi Payload Traffic Selector**

<table>
<thead>
<tr>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH2’s Global Address on Link X</td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH2’s Global Address on Link X</td>
</tr>
</tbody>
</table>

**TSr Payload Traffic Selector**

<table>
<thead>
<tr>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix B:0000:0000:0000:0000:0000:0000:0000</td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix B:ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>

Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #9</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #9</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #9</td>
</tr>
</tbody>
</table>

**TSi Payload Traffic Selector**

<table>
<thead>
<tr>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH3’s Global Address on Link X</td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH3’s Global Address on Link X</td>
</tr>
</tbody>
</table>

**TSr Payload Traffic Selector**

<table>
<thead>
<tr>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>Prefix B:0000:0000:0000:0000:0000:0000:0000</td>
</tr>
<tr>
<td>Ending Address</td>
<td>Prefix B:ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>
**Part A (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request including ICMPv6 (58) as IP Protocol ID value in Traffic Selector Payload.
6. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**

The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**

The NUT transmits a CREATE_CHILD_SA response including a Notify payload of type TS_UNACCEPTABLE.

**Possible Problems:**

- None.
Test IKEv2.SGW.R.1.1.7.3: Narrowing Traffic Selectors

Purpose:

To verify an IKEv2 device allows the responder to choose a subset of the traffic proposed by the initiator.

References:

- [RFC4306] - Section 2.8
- [RFC4718] - Section 4.10

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration except Traffic Selector. Traffic Selector should be configured as following.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Address Range</td>
<td>Next Layer Protocol</td>
</tr>
<tr>
<td>Inbound</td>
<td>TH2</td>
<td>ANY</td>
</tr>
<tr>
<td>Outbound</td>
<td>NUT</td>
<td>ANY</td>
</tr>
</tbody>
</table>

The other packets are allowed to BYPASS IPsec protection.

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #5</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #5</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #5</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See below</td>
</tr>
</tbody>
</table>

**TSi Payload**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH2’s Global Address on Link X</td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH2’s Global Address on Link X</td>
<td></td>
</tr>
</tbody>
</table>

**TSr Payload**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

908
IPv6 Ready Logo Program IKEv2

Start Port 0
End Port 65535
Starting Address TH3’s Global Address on Link X
Ending Address TH3’s Global Address on Link X

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPv6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Starting Address</td>
<td>TH1’s Global Address on Link A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ending Address</td>
<td>TH1’s Global Address on Link A</td>
</tr>
</tbody>
</table>

○ Packet #5: ICMPv6 Echo Request

IPv6 Header | Same as the Common Packet #21
ESP | Same as the Common Packet #21
IPv6 Header | Source Address TH3’s Global Address
Destination Address | TH1’s Global Address
ICMPv6 Header | Same as the Common Packet #21

○ Packet #6: ICMPv6 Echo Request

IPv6 Header | Source Address TH1’s Global Address
Destination Address | TH3’s Global Address
ICMPv6 Header | Same as the Common Packet #25

Part A (BASIC)

1. TN1 sends an IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. TN1 sends an IKE_SA_INIT request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request packet to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply packet to TH2.
8. Observe the messages transmitted on Link A.
9. TH3 transmits an Echo Request to TH1.
10. Observe the messages transmitted on Link B.
11. TH1 transmits an Echo Request to TH3.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. The Traffic Selector is narrowed to allow the traffic from/to TH2.

Step 6: Judgment #3
The NUT forwards an Echo Request.

**Step 8: Judgment #4**
The NUT forwards an Echo Request with IPsec ESP using corresponding algorithms.

**Step 10: Judgment #5**
The NUT never forwards an Echo Request.

**Step 12: Judgment #6**
The NUT forwards an Echo Request without IPsec ESP.

**Possible Problems:**

- Because the destination address of Echo Request is the TN itself, TN may respond to Echo Request automatically. In that case, TH2 can send Echo Reply to TH1 instead of sending Echo Request.
Group 1.8. Error Handling

Test IKEv2.SGW.R.1.1.8.1: INVALID_IKE_SPI

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.R.1.1.8.2: INVALID_SYNTAX

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.R.1.1.8.3: INVALID_SELECTORS

This test case was deleted at revision 1.1.0.
Group 1.10 Authentication of the IKE_SA

Test IKEv2.SGW.R.1.1.10.1: Sending Certificate Payload

Purpose:
To verify an IKEv2 device handles a CERTREQ payload and transmits a CERT payload properly.

References:
- [RFC 4306] - Sections 1.2 and 3.8

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Part</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>NUT’s global address on Link A</td>
</tr>
<tr>
<td>Part B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>nut.example.com</td>
</tr>
<tr>
<td>Part C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:nut@example.com">nut@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

- Packet #1
  See Common Packet #1

- Packet #2: IKE_AUTH request
  IPv6 Header: Same as the Common Packet #5
UDP Header | Same as the Common Packet #5
---|---
IKEv2 Header | Same as the Common Packet #5
E Payload | Same as the Common Packet #5
IDi Payload | Next Payload | 38 (CERTREQ)
Oter fields are same as the Common Packet #5
CERTREQ Payload | See below
AUTH Payload | Same as the Common Packet #5
SA Payload | Same as the Common Packet #5
TSi Payload | Same as the Common Packet #5
TSr Payload | Same as the Common Packet #5
CERTREQ Payload | Next Payload | 39 (AUTH)
Critical | 0
Reserved | 0
Payload Length | Any
Certificate Encoding | 4 (X.509 Certificate - Signature)
Certificate Authority | any

**Part A: ID_IPV6_ADDR (ADVANCED)**
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with a CERTREQ payload to the NUT.
4. Observe the messages transmitted on Link A.

**Part B: ID_FQDN (ADVANCED)**
5. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
6. Observe the messages transmitted on Link A.
7. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with a CERTREQ payload to the NUT.
8. Observe the messages transmitted on Link A.

**Part C: ID_RFC822_ADDR (ADVANCED)**
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with a CERTREQ payload to the NUT.
12. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response. The response includes an ID payload with ID_IPV6_ADDR and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Part B**

**Step 6: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 8: Judgment #2**
The NUT transmits an IKE_AUTH response. The response includes an ID payload with ID_FQDN and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH response. The response includes an ID payload with ID_RFC822_ADDR and a CERT payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding and the NUT’s certificate as Certificate Data.

**Possible Problems:**

- None.
Test IKEv2.SGW.R.1.1.10.2: Sending Certificate Request Payload

Purpose:
To verify an IKEv2 device properly transmits CERTREQ payload.

References:
- [RFC 4306] - Sections 1.2 and 3.7

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Remote</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>TN1’s global address on Link A</td>
</tr>
<tr>
<td>Part B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>tn.example.com</td>
</tr>
<tr>
<td>Part C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:tn@example.com">tn@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

Part A: ID_IPV6_ADDR (ADVANCED)

Part B: ID_FQDN (ADVANCED)

Part C: ID_RFC822_ADDR (ADVANCED)

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Part B

Step 4: Judgment #1
The NUT transmits an IKE_SA_INIT response with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Part C

Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response with a CERTREQ payload which contains 4 (X.509 Certificate - Signature) as Certificate Encoding.

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.10.3: RSA Digital Signature

Purpose:

To verify an IKEv2 device authenticates the corresponding node by RSA Digital Signature.

References:

- [RFC 4306] - Sections 1.2 and 3.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Remote</th>
<th>Authentication Method</th>
<th>ID Type</th>
<th>ID Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part A</td>
<td>X.509 Certificate - Signature</td>
<td>ID_IPV6_ADDR</td>
<td>TN1's global address on Link A</td>
</tr>
<tr>
<td>Part B</td>
<td>X.509 Certificate - Signature</td>
<td>ID_FQDN</td>
<td>tn.example.com</td>
</tr>
<tr>
<td>Part C</td>
<td>X.509 Certificate - Signature</td>
<td>ID_RFC822_ADDR</td>
<td><a href="mailto:tn@example.com">tn@example.com</a></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet</th>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
<td></td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #19</td>
<td></td>
</tr>
</tbody>
</table>

- Packet #2: IKE_AUTH request
IPv6 Header: Same as the Common Packet #5
UDP Header: Same as the Common Packet #5
IKEv2 Header: Same as the Common Packet #5
E Payload: Same as the Common Packet #5
IDi Payload: Next Payload 37 (CERT5)
Other fields are same as the Common Packet #5
CERT Payload: See below
AUTH Payload: Same as the Common Packet #5
N Payload: Same as the Common Packet #5
SA Payload: Same as the Common Packet #5
TSi Payload: Same as the Common Packet #5
TSr Payload: Same as the Common Packet #5

CERT Payload:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>39 (AUTH)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>Any</td>
</tr>
<tr>
<td>Certificate Encoding</td>
<td>4 (X.509 Certificate – Signature)</td>
</tr>
<tr>
<td>Certificate Data</td>
<td>any</td>
</tr>
</tbody>
</table>

Part A: ID_IPV6_ADDR (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with an IDi payload as described above and a CERT payload to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.

Part B: ID_FQDN (ADVANCED)
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with an IDi payload as described above and a CERT payload to the NUT.
12. Observe the messages transmitted on Link A.
13. TH2 transmits an Echo Request to TH1.
14. Observe the messages transmitted on Link B.
15. TH1 transmits an Echo Reply to TH2.
16. Observe the messages transmitted on Link A.

Part C: ID_RFC822_ADDR (ADVANCED)
17. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
18. Observe the messages transmitted on Link A.
19. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with an IDi payload as described above and a CERT payload to the NUT.
20. Observe the messages transmitted on Link A.
21. TH2 transmits an Echo Request to TH1.
22. Observe the messages transmitted on Link B.
23. TH1 transmits an Echo Reply to TH2.
24. Observe the messages transmitted on Link A.

Observable Results:
IPv6 FORUM TECHNICAL DOCUMENT

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Part B

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 14: Judgment #3
The NUT forwards an Echo Request.

Step 16: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Part C

Step 18: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 20: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 22: Judgment #3
The NUT forwards an Echo Request.

Step 24: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Possible Problems:
• None.
Test IKEv2.SGW.R.1.1.10.4: HEX string PSK

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key.

References:

- [RFC 4306] - Sections 2.15

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the following IKE peer configuration.

<table>
<thead>
<tr>
<th>Authentication Key Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
</tr>
<tr>
<td>0xabadcafeabadcafeabadcafeabadcafe (128 bit binary string)</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Possible Problems:

- None.
Group 1.11 Invalid values

Test IKEv2.SGW.R.1.1.11.1: Non zero RESERVED fields in IKE_SA_INIT request

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

---

**Packet #1** See Common Packet #1
All RESERVED fields are set to one.

**Part A (BASIC)**
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Possible Problems:**
• None.
Test IKEv2.SGW.R.1.1.11.2: Non zero RESERVED fields in IKE_AUTH request

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td></td>
<td>All RESERVED fields are set to one.</td>
</tr>
</tbody>
</table>

Part A (BASIC)
1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.
Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Possible Problems:

• None.
Test IKEv2.SGW.R.1.1.11.3: Version bit is set

Purpose:

To verify an IKEv2 device ignores the content of Version in IKE messages.

References:

- [RFC 4306] - Sections 3.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version bit is set to one.</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)

1. TN starts to negotiate with NUT by sending IKE_SA_INIT request whose Version bit is set to one.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.11.4: Response bit is set

Purpose:

To verify an IKEv2 device ignores an IKE request message whose Response bit is set.

References:

- [RFC 4306] - Sections 2.21

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response bit is set to one.</td>
<td></td>
</tr>
</tbody>
</table>

Part A (BASIC)

1. TN starts to negotiate with NUT by sending IKE_SA_INIT request whose Response bit is set to one.
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1

The NUT never responds with an IKE_SA_INIT response to an IKE_SA_INIT request from the TN1.

Possible Problems:

- None.
Test IKEv2.SGW.R.1.1.11.5: Unrecognized Notify Message Type

Purpose:

To verify an IKEv2 device ignores the unrecognized Notify Message Type in IKE messages.

References:

- [RFC 4306] - Sections 3.10.1

Test Setup:

- **Network Topology**
  Connect the devices according to the Common Topology.
- **Configuration**
  In each part, configure the devices according to the Common Configuration.
- **Pre-Sequence and Cleanup Sequence**
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet</th>
<th>See Common Packet #1</th>
<th>See below</th>
<th>See Common Packet #21</th>
<th>See Common Packet #25</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #1</td>
<td>See Common Packet #1</td>
<td>Packet #2 See below</td>
<td>Packet #3 See Common Packet #21</td>
<td>Packet #4 See Common Packet #25</td>
</tr>
</tbody>
</table>

Packet #2: IKE_AUTH request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>All fields are same as Common Packet #5</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>All fields are same as Common Packet #5</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>All fields are same as Common Packet #5</td>
</tr>
<tr>
<td>E Payload</td>
<td>All fields are same as Common Packet #5</td>
</tr>
</tbody>
</table>
Part A: Unrecognized Notify Message Type of error 16383 (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with a Notify payload of unrecognized Notify Message Type value (16383) to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.

Part B: Unrecognized Notify Message Type of status 65535 (BASIC)
9. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request with a Notify payload of unrecognized Notify Message Type value (65535) to the NUT.
12. Observe the messages transmitted on Link A.
13. TH2 transmits an Echo Request to TH1.
14. Observe the messages transmitted on Link B.
15. TH1 transmits an Echo Reply to TH2.
16. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.
Part B

Step 10: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 12: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 14: Judgment #3
The NUT forwards an Echo Request.

Step 16: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using ENCR_3DES and AUTH_HMAC_SHA1_96.

Possible Problems:

- None.
Group 2. The CREATE_CHILD_SA Exchange

Group 2.1. Header and Payload Formats

Test IKEv2.SGW.R.1.2.1.1: Receipt of CREATE_CHILD_SA request

Purpose:

To verify an IKEv2 device transmits a CREATE_CHILD_SA response using properly Header and Payloads format

References:

- [RFC 4306] - Sections 1.1.2, 1.2 and 3.3.2
- [RFC 4307] - Sections 3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #</th>
<th>See Common Packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #1</td>
<td>See Common Packet #1</td>
</tr>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #15</td>
</tr>
</tbody>
</table>

Part A: IKE Header Format (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
6. Observe the messages transmitted on Link A.

Part B: Encrypted Payload Format (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
12. Observe the messages transmitted on Link A.

Part C: SA Payload Format (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
18. Observe the messages transmitted on Link A.

Part D: Nonce Payload Format (BASIC)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
24. Observe the messages transmitted on Link A.

Part E: TSi Payload Format (BASIC)
25. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
26. Observe the messages transmitted on Link A.
27. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH request to the NUT.
28. Observe the messages transmitted on Link A.
29. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
30. Observe the messages transmitted on Link A.

Part F: TSr Payload Format (BASIC)
31. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
32. Observe the messages transmitted on Link A.
33. After a reception of IKE_SA_INIT response from the NUT, TN1 transmits IKE_AUTH
34. Observe the messages transmitted on Link A.
35. After reception of IKE_AUTH response from the NUT, TN1 transmits CREATE_CHILD_SA request to the NUT to rekey CHILD_SAs.
36. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including properly formatted IKE Header containing following values:

![Figure 160 Header format](image)

- An IKE_SA Initiator’s SPI field set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field set to Encrypted Payload (46).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to CREATE_CHILD_SA (36).
- A Flags field set to (00000100)₂ = (4)₁₀.
- A Message ID field set to the same value as corresponding IKEv2 request message’s Message ID.
- A Length field set to the length of the message (header + payloads) in octets.

Part B

Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 10: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 12: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including properly formatted Encrypted Payload containing following values:

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Checksum Data.
- An Initialization Vector field set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
- An Encrypted IKE Payloads field set to subsequent payloads encrypted by ENCR_3DES.
- A Padding field set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
- A Pad Length field set to the length of the Padding field.
- An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

Part C

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**

The NUT transmits a CREATE_CHILD_SA response including properly formatted SA Payload containing following values (refer following figures):

**Figure 162 SA Payload contents**

- A Next Payload field set to Nr Payload (40).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.

A Proposals field set to following.
Proposal #1
- A 0 or 2 field set to zero (last).
- A RESREVD field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field set to 1.
- A Protocol ID field set to ESP (3).
- A SPI Size field set to 4.
- A # of Transforms field set to 3.
- A SPI field set to the sending entity’s SPI (4 octets value)

Transform field set to following (There are 3 Transform Structures).

Transform #1
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ENCR_3DES.
- A Transform Type field set to ENCR (1).
- A RESERVED field set to zero.
- A Transform ID set to ENCR_3DES (3).

Transform #2
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including
Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.

- A Transform Type field set to INTEG (3).
- A RESERVED field set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
- A Transform Type field set to ESN (5).
- A RESERVED field set to zero.
- A Transform ID set to No Extended Sequence Numbers (0).

**Part D**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 22: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 24: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including properly formatted Nonce Payload containing following values:

![Figure 166 Nonce Payload format](image)

- A Next Payload field set to TSi Payload (44).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Nonce Data field set to random data generated by the transmitting entity. The size of the Nonce must between 16 and 256 octets.

**Part E**

**Step 26: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 28: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 30: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including properly formatted TSi Payload containing following values:

- A Next Payload field set to TSr Payload (45).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to 1.
- A RESERVED field set to zero.

Traffic Selectors field set to following.

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to Prefix Y.
- A Ending Address field set to greater than or equal to Prefix Y.
Part G

Step 32: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 34: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 36: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including properly formatted TSr Payload containing following values:

```
Figure 169 TSr Payload format
```
- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to the number of actual traffic selectors.
- A RESERVED field set to zero.

Traffic Selectors field set following:

```
Figure 170 Traffic Selector
```
- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
• A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
• A Start Port field set to zero.
• An End Port field set to 65535.
• A Starting Address field set to less than or equal to Prefix B.
• An Ending Address field set to less than or equal to Prefix B.

Possible Problems:

• CREATE_CHILD_SA response has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

```
CREATE_CHILD_SA response has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.
```

• Each of transforms can be located in the any order.
Group 2.2. Use of Retransmission Timers

Test IKEv2.SGW.R.1.2.2.1: Receipt of CREATE_CHILD_SA requests

Purpose:
To verify an IKEv2 device retransmits CREATE_CHILD_SA request using properly Header and Payloads format

References:
- [RFC 4306] - Sections 2.1, 2.2 and 2.4

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Part A: (BASIC)
1. TN1 transmits IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits CREATE_CHILD_SA request.
6. Observe the messages transmitted on Link A.
7. Observe the messages transmitted on Link A.
8. TN1 transmits the same CREATE_CHILD_SA request packet as Step 5.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 7: Judgment #4
The NUT never retransmits a CREATE_CHILD_SA response which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

Step 9: Judgment #5
The NUT retransmits a CREATE_CHILD_SA response which has the same Message ID value as the previous CREATE_CHILD_SA request’s Message ID value in IKE Header.

Possible Problems:

• none
Group 2.3. State Synchronization and Connection Timeouts

Test IKEv2.SGW.R.1.2.3.1: Receiving Delete Payload for Multiple CHILD_SA

Purpose:

To verify an IKEv2 device transmits a Delete Payload, when CHILD_SAs are deleted.

References:

- [RFC 4306] - Sections 2.4 and 3.11

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See below</td>
</tr>
</tbody>
</table>

- Packet #2: IKE_AUTH request
### IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #5</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #5</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #5</td>
</tr>
<tr>
<td><strong>Other fields</strong></td>
<td>Same as the Common Packet #5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>6 (TCP)</td>
<td>65535</td>
<td></td>
<td></td>
<td>Prefix Y:0000:0000:0000:0000</td>
<td>Prefix Y:ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>T Sr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>6 (TCP)</td>
<td>65535</td>
<td></td>
<td></td>
<td>Prefix B:0000:0000:0000:0000</td>
<td>Prefix B:ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>

- Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #9</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #9</td>
</tr>
<tr>
<td>T Sr Payload</td>
<td>Other fields are same as the Common Packet #9</td>
</tr>
<tr>
<td><strong>Other fields</strong></td>
<td>Same as the Common Packet #9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>58 (ICMPv6)</td>
<td>65535</td>
<td></td>
<td></td>
<td>Prefix Y:0000:0000:0000:0000</td>
<td>Prefix Y:ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>T Sr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>58 (ICMPv6)</td>
<td>65535</td>
<td></td>
<td></td>
<td>Prefix B:0000:0000:0000:0000</td>
<td>Prefix B:ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>
Packet #4: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
</tbody>
</table>

**E Payload**

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>42 (Delete)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>16</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of SPIs</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Security Parameter Index(es) (SPI)</th>
<th>SPI negotiated by Initial Exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPI negotiated by CREATE_CHILD_SA exchange</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. TN starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN transmits a CREATE_CHILD_SA request to establish a new CHILD_SA to the NUT.
6. Observe the messages transmitted on Link A.
7. TN transmits an INFORMATIONAL request with a Delete payload including the first negotiated CHILD_SA’s inbound SPI and the second negotiated CHILD_SA’s inbound SPI.
8. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 8: Judgment #4**
The NUT transmits an INFORMATIONAL response with delete payload for SPIs which are negotiated by Initial Exchange and CREATE_CHILD_SA exchange.

**Possible Problems:**

- INFORMATIONAL response from NUT may not contain Delete Payload by implementation policy. This behavior is defined at section 1.4 in RFC 4306 as an exception.
Group 2.4. Cryptographic Algorithm Negotiation

Test IKEv2.SGW.R.1.2.4.1: Sending NO_PROPOSAL_CHOSEN

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with an unacceptable SA payload.

References:

- [RFC 4306] - Sections 2.7 and 3.10.1
- [RFC 4718] - Sections 2.1 and 2.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>
Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #15</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See below</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>TSI Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>TSR Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td># of Transforms</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI any</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>12 (AES_CBC)</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>5 (AES_CBC,96)</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>5 (ESN)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>1 (ESN)</td>
<td></td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request to rekey the established CHILD_SAs to the NUT. The CREATE_CHILD_SA request includes a SA payload with a proposal unaccepted by the NUT.
6. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT does not transmit a CREATE_CHILD_SA response or transmits a CREATE_CHILD_SA response including a Notify payload of type NO_PROPOSAL_CHOSEN.

**Possible Problems:**

- None.
Group 2.5. Rekeying CHILD_SA Using a CREATE_CHILD_SA exchange

Test IKEv2.SGW.R.1.2.5.1: Close the replaced CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the CREATE_CHILD_SA Exchanges to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #6: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td>Delete Payload</td>
<td></td>
</tr>
<tr>
<td>Next Payload</td>
<td>42 (Delete)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>12</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td># of SPis</td>
<td>1</td>
</tr>
<tr>
<td>Security Parameter Index(es) (SPI)</td>
<td>SPI negotiated by Initial Exchange</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request including a Delete payload with the old CHILD_SA’s SPI value to the NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.
Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 12: Judgment #6
The NUT transmits an INFORMATIONAL response including a Delete payload with the old CHILD_SA’s SPI value to the TN1.

Possible Problems:

- None.
**Test IKEv2.SGW.R.1.2.5.2: Use of the new CHILD_SA**

**Purpose:**
To verify an IKEv2 device properly recognizes the lifetime of CHILD_SAs.

**References:**
- [RFC 4306] - Sections 2.8

**Test Setup:**
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IKEv2 on the NUT is disabled.</td>
</tr>
<tr>
<td>2</td>
<td>Connect the devices according to the Common Topology.</td>
</tr>
<tr>
<td>3</td>
<td>In each part, configure the devices according to the Common Configuration.</td>
</tr>
<tr>
<td>4</td>
<td>IKEv2 on the NUT is disabled after each part.</td>
</tr>
<tr>
<td>Packet #6: INFORMATIONAL request</td>
<td>Source Address</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>IPv6 Header</td>
<td>Source Port</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Source Port</td>
</tr>
<tr>
<td></td>
<td>Destination Port</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>IKE_SA Initiator’s SPI</td>
</tr>
<tr>
<td></td>
<td>IKE_SA Responder’s SPI</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Major Version</td>
</tr>
<tr>
<td></td>
<td>Minor Version</td>
</tr>
<tr>
<td></td>
<td>Exchange Type</td>
</tr>
<tr>
<td></td>
<td>X (bits 0–2 of Flags)</td>
</tr>
<tr>
<td></td>
<td>I (bit 3 of Flags)</td>
</tr>
<tr>
<td></td>
<td>V (bit 4 of Flags)</td>
</tr>
<tr>
<td></td>
<td>R (bit 5 of Flags)</td>
</tr>
<tr>
<td></td>
<td>X (bits 6–7 Flags)</td>
</tr>
<tr>
<td></td>
<td>Message ID</td>
</tr>
<tr>
<td></td>
<td>Length</td>
</tr>
<tr>
<td>E Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
</tr>
<tr>
<td></td>
<td>Initialization Vector</td>
</tr>
<tr>
<td></td>
<td>Encrypted IKE Payloads</td>
</tr>
<tr>
<td></td>
<td>Padding</td>
</tr>
<tr>
<td></td>
<td>Pad Length</td>
</tr>
<tr>
<td></td>
<td>Integrity Checksum Data</td>
</tr>
<tr>
<td>D Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
</tr>
<tr>
<td></td>
<td># of SPIs</td>
</tr>
<tr>
<td></td>
<td>Security Parameter Index</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request using the first negotiated algorithms to the NUT.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request with a Delete payload to the NUT.
12. Observe the messages transmitted on Link A.
13. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request using the second negotiated algorithms to the NUT.
14. Observe the messages transmitted on Link A.
15. TH1 transmits an Echo Reply to TH2.
16. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 12: Judgment #6
The NUT transmits an INFORMATIONAL response with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the NUT’s inbound SPI value to be deleted as SPI value.

Step 14: Judgment #7
The NUT forwards an Echo Request.

Step 16: Judgment #8
The NUT forwards an Echo Reply with IPsec ESP using the second negotiated algorithms.

Possible Problems:

- none
Test IKEv2.SGW.R.1.2.5.3: Receiving Multiple Transform

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with multiple transforms to rekey CHILD_SA.

References:

- [RFC 4306] - Sections 2.7, 2.8 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

From part A to part C, TN1 transmits a CREATE_CHILD_SA request including a SA payload which contains the transforms as follows:

<table>
<thead>
<tr>
<th>Part</th>
<th>CREATE_CHILD_SA exchanges Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Encryption</td>
</tr>
<tr>
<td>Part A</td>
<td>ENCR_3DES</td>
</tr>
<tr>
<td>Part B</td>
<td>ENCR_3DES</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 958 IPv6 Ready Logo Program IKEv2
Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #15</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>TSI Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
</tbody>
</table>

Proposal #1 SA Proposal

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>Reserved</th>
<th>Proposal Length</th>
<th>Proposal ID</th>
<th>SPI Size</th>
<th># of Transforms</th>
<th>SPI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 (ESP)</td>
<td>4</td>
<td>4</td>
<td>Any</td>
</tr>
</tbody>
</table>

SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>According to above configuration</td>
<td>0</td>
<td>3 (3DES)</td>
</tr>
</tbody>
</table>

SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 (more)</td>
<td>0</td>
<td>8</td>
<td>1 (ENC)</td>
<td>0</td>
<td>3 (HMAC_SHA1_96)</td>
</tr>
</tbody>
</table>

SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Transform ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (last)</td>
<td>0</td>
<td>8</td>
<td>5 (ESN)</td>
<td>0</td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

Part A: Multiple Encryption Algorithms (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

Part B: Multiple Integrity Algorithms (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

Part C: Multiple Extended Sequence Numbers (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

Observable Results:

Part A
Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part B
Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 10: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 12: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part C
Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 18: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

- none
Test IKEv2.SGW.R.1.2.5.4: Receiving Multiple Proposal

Purpose:
To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with multiple transforms to rekey CHILD_SA.

References:
- [RFC 4306] - Sections 2.7, 2.8 and 3.3

Test Setup:
- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Proposal ID</th>
<th>Encryption</th>
<th>Integrity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal #1</td>
<td>ENCR_AES_CBC</td>
<td>AUTH HMAC_SHA1 96</td>
</tr>
<tr>
<td>Proposal #2</td>
<td>ENCR_3DES</td>
<td>AUTH HMAC_SHA1 96</td>
</tr>
</tbody>
</table>

TN1 transmits a CREATE_CHILD_SA request including a SA payload which contains the two proposals as follows:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>
### IPv6 FORUM TECHNICAL DOCUMENT

<table>
<thead>
<tr>
<th>Proposal</th>
<th>ESP</th>
<th>ENCR_3DES</th>
<th>AUTH_HMAC_SHA1_96</th>
<th>ESN</th>
<th>No ESN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal #1</td>
<td>ESP</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td><strong>ESN</strong></td>
<td></td>
</tr>
<tr>
<td>Proposal #2</td>
<td>ESP</td>
<td>ENCR_3DES</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>No ESN</td>
<td></td>
</tr>
</tbody>
</table>

- Packet #3: CREATE_CHILD_SA request

```
| IPv6 Header | Same as the Common Packet #15 |
| UDP Header  | Same as the Common Packet #15 |
| IKEv2 Header| Same as the Common Packet #15 |
| E Payload   | Same as the Common Packet #15 |
| IDi Payload | Same as the Common Packet #15 |
| AUTH Payload| Same as the Common Packet #15 |
| N Payload   | Same as the Common Packet #15 |
| SA Payload  | Other fields are same as the Common Packet #15 |
| TSI Payload | Same as the Common Packet #15 |
| TSr Payload | Same as the Common Packet #15 |

**Proposal #1**

- SA Proposal
- Next Payload 2 (more)
- Reserved 0
- Proposal Length 40
- Proposal # 1
- Proposal ID 3 (ESP)
- SPI Size 4
- # of Transforms 4
- SPI Any

- SA Transform
  - Next Payload 3 (more)
  - Reserved 0
  - Transform Length 8
  - Transform Type According to above configuration
  - Reserved 0
  - Transform ID According to above configuration

- SA Transform
  - Next Payload 0 (last)
  - Reserved 0
  - Transform Length 8
  - Transform Type According to above configuration
  - Reserved 0
  - Transform ID According to above configuration

**Proposal #2**

- SA Proposal
- Next Payload 0 (last)
- Reserved 0
- Proposal Length 40
- Proposal # 2
- Proposal ID 3 (ESP)
- SPI Size 4
- # of Transforms 4
- SPI Any

- SA Transform
  - Next Payload 3 (more)
  - Reserved 0
  - Transform Length 8
  - Transform Type 1 (ENCR)
  - Reserved 0
  - Transform ID 3 (3DES)

- SA Transform
  - Next Payload 3 (more)
  - Reserved 0
  - Transform Length 8
```
<table>
<thead>
<tr>
<th>Transform Type</th>
<th>3 (INTEG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>5 (ESN)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>0 (No ESN)</td>
</tr>
</tbody>
</table>

**Part A: Multiple Encryption Algorithms (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

**Part B: Multiple Integrity Algorithms (BASIC)**

7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

**Part C: Multiple Extended Sequence Numbers (BASIC)**

13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**

The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part B

Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 10: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 12: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Part C

Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 18: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

- none
Test IKEv2.SGW.R.1.2.5.5: Perfect Forward Secrecy

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA exchange when Perfect Forward Secrecy enables.

References:

- [RFC 4306] - Sections 2.12

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
Packet #8: See Common Packet #25

Packet #5: CREATE_CHILD_SA response

<table>
<thead>
<tr>
<th>Payload</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>Ni Payload</td>
<td>Next Payload 34 (KE)</td>
</tr>
<tr>
<td>KEi Payload</td>
<td>Next Payload 44 (TSi)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
</tbody>
</table>
Packet #6: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td>Delete Payload</td>
<td>Next Payload 42 (Delete)</td>
</tr>
<tr>
<td></td>
<td>Next Payload 0 (last)</td>
</tr>
<tr>
<td></td>
<td>Critical 0</td>
</tr>
<tr>
<td></td>
<td>Reserved 0</td>
</tr>
<tr>
<td></td>
<td>Payload Length 12</td>
</tr>
<tr>
<td></td>
<td>Protocol ID 3 (ESP)</td>
</tr>
<tr>
<td></td>
<td>SPI Size 4</td>
</tr>
<tr>
<td></td>
<td># of SPIs 1</td>
</tr>
<tr>
<td></td>
<td>Security Parameter Index(es) (SPI) SPI negotiated by Initial Exchange</td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request using the first negotiated algorithms to the NUT.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request with a Delete payload to the NUT.
12. Observe the messages transmitted on Link A.
13. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request using the second negotiated algorithms to the NUT.
14. Observe the messages transmitted on Link B.
15. TH1 transmits an Echo Reply to TH2.
16. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.
Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 10: Judgment #5
The NUT forwards an Echo Request.

Step 10: Judgment #5
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 12: Judgment #6
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 12: Judgment #6
The NUT transmits an INFORMATIONAL response with a Delete payload. The Delete payload includes 3 (ESP) as Protocol ID, 4 as SPI Size and the NUT’s inbound SPI value to be deleted as SPI value.

Step 14: Judgment #7
The NUT forwards an Echo Request.

Step 16: Judgment #8
The NUT forwards an Echo Reply with IPsec ESP using the second negotiated algorithms.

Possible Problems:

- none
Test IKEv2.SGW.R.1.2.5.6: Use of the old CHILD_SA

Purpose:

To verify an IKEv2 device properly handles new CHILD_SA and old CHILD_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #3</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
</tbody>
</table>
Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request using the first negotiated algorithms to the NUT.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
10. Observe the messages transmitted on Link A.
11. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request using the first negotiated algorithms again.
12. Observe the messages transmitted on Link B.
13. TH1 transmits an Echo Reply to TH2.
14. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 12: Judgment #6
The NUT forwards an Echo Request.

Step 14: Judgment #7
The NUT forwards an Echo Reply with IPsec ESP. The NUT can use both the first CHILD_SA and the new CHILD_SA.

Possible Problems:

- none
Group 2.6. Rekeying IKE_SAs Using a CREATE_CHILD_SA exchange

Test IKEv2.SGW.R.1.2.6.1: Sending CREATE_CHILD_SA response

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8 and 2.18

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>See Common Packet #1</td>
</tr>
<tr>
<td>#2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>#3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>#4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>#5</td>
<td>See Common Packet #11</td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request including a SA payload. The proposal in the SA payload contains 1 (IKE) in the Protocol ID field, 8 in the SPI size field and the rekeyed IKE_SA Initiator’s SPI value.
10. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the proposal in the SA payload Response includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA Responder’s SPI value in the SPI field.

Possible Problems:

- none
Test IKEv2.SGW.R.1.2.6.2: Receipt of cryptographically valid message on the old SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>
Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request including a SA payload. A proposal in the SA payload contains 1 (IKE) in the Protocol ID field, 8 in the SPI size field and the rekeyed IKE_SA Initiator’s SPI value.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request with no payloads protected by the old IKE_SA.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA Responder’s SPI value in the SPI field.

Step 12: Judgment #6
The NUT responds with an INFORMATIONAL response with no payloads protected by the old IKE_SA.

Possible Problems:

- none
Test IKEv2.SGW.R.1.2.6.3: Receipt of cryptographically valid message on the new SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request including a SA payload. A proposal in the SA payload contains 1 (IKE) in the Protocol ID field, 8 in the SPI size field and the rekeyed IKE_SA Initiator’s SPI value.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request with no payloads protected by the new IKE_SA and the Message ID field in the IKE header is zero.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA Responder’s SPI value in the SPI field.

Step 12: Judgment #6
The NUT responds with an INFORMATIONAL response with no payloads protected by the new IKE_SA and the Message ID field in the IKE header is zero.

Possible Problems:

- none
Test IKEv2.SGW.R.1.2.6.4: Close the replaced IKE_SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8
- [RFC 4718] - Sections 5.8 and 5.11

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

```
Packet #1
```

See Common Packet #1
Packet #6: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td>Delete Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
</tr>
<tr>
<td></td>
<td># of SPIs</td>
</tr>
<tr>
<td></td>
<td>Security Parameter Index(es) (SPI)</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request to rekey IKE_SA. A proposal in the SA payload contains 1 (IKE) in the Protocol ID field, 8 in the SPI size field and the rekeyed IKE_SA Initiator’s SPI value.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request with a Delete payload which has 1 (IKE_SA) in the Protocol ID field, zero in the SPI Size field and zero in the # of SPIs field.
12. Observe the messages transmitted on Link A.
13. TH2 transmits an Echo Request to TH1. TN1 forwards an Echo Request with IPsec ESP with corresponding algorithms inherited from the replaced IKE_SA.
14. Observe the messages transmitted on Link B.
15. TH1 transmits an Echo Reply to TH2.
16. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**

The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.
Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA Responder’s SPI value in the SPI field.

Step 12: Judgment #6
The NUT responds with an INFORMATIONAL response with no payloads.

Step 14: Judgment #3
The NUT forwards an Echo Request.

Step 16: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms inherited from the replaced IKE_SA.

Possible Problems:

- none
Test IKEv2.SGW.R.1.2.6.5: Receiving Multiple Transform

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with multiple transform to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.7, 2.8 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Part</th>
<th>IKE_SA_INIT exchanges Algorithms</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ENCR_AES_CBC  ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>ENCR_3DES</td>
<td>PRF_AES128_CBC  PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_AES_XCBC_96</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
</tbody>
</table>

From part A to part D, TN1 transmits an IKE_SA_INIT request including a SA payload which contains the transforms as follows:
**Packet #3 CREATE_CHILD_SA request**

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #11</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #11</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #11</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the common packet #11</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See SA Table below</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #1</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal Length</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protocol ID</td>
<td>1 (IKE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>According to above configuration</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>According to above configuration</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>3 (3DES)</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>2 (PRF)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>2 (HMAC_SHA1)</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>2 (HMAC_SHA1_96)</td>
<td></td>
</tr>
<tr>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0 (last)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Length</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform Type</td>
<td>4 (D-H)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transform ID</td>
<td>2 (1024 MODP Group)</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: Multiple Encryption Algorithms (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.
Part B: Multiple Pseudo Random Function (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

Part C: Multiple Integrity Algorithm (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

Part D: Multiple D-H Group (BASIC)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
24. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Part B

Step 8: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.
Step 10: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 12: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed
algorithms.

Part C
Step 14: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted
algorithms.

Step 16: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 18: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed
algorithms.

Part D
Step 20: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted
algorithms.

Step 22: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 24: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed
algorithms.

Possible Problems:

• none
Test IKEv2.SGW.R.1.2.6.6: Receiving Multiple Proposal

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with multiple proposal to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.7, 2.8 and 3.3

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

TN1 transmits a CREATE_CHILD_SA request including a SA payload which contains the two proposals as follows:

<table>
<thead>
<tr>
<th>Part</th>
<th>Proposal ID</th>
<th>IKE</th>
<th>Encryption</th>
<th>PRF</th>
<th>Integrity</th>
<th>D-H Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Proposal #1</td>
<td>IKE</td>
<td>ENCR_AES_CBC</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>A</td>
<td>Proposal #2</td>
<td>IKE</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>B</td>
<td>Proposal #1</td>
<td>IKE</td>
<td>ENCR_3DES</td>
<td>PRF_AES128_CBC</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
<tr>
<td>B</td>
<td>Proposal #2</td>
<td>IKE</td>
<td>ENCR_3DES</td>
<td>PRF_HMAC_SHA1</td>
<td>AUTH_HMAC_SHA1_96</td>
<td>Group 2</td>
</tr>
</tbody>
</table>
### Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #11</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #11</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #11</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the common packet #11</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See SA Table below</td>
</tr>
<tr>
<td>Ni, Nr Payload</td>
<td>Same as the Common Packet #11</td>
</tr>
</tbody>
</table>

#### Proposal #1

<table>
<thead>
<tr>
<th>Proposal</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>2 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal Length</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protocol ID</td>
<td>1 (IKE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td># of Transforms</td>
<td>5</td>
</tr>
</tbody>
</table>

#### SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>1 (ENC)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

#### Proposal #2

<table>
<thead>
<tr>
<th>Proposal</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal Length</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protocol ID</td>
<td>1 (IKE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td># of Transforms</td>
<td>5</td>
</tr>
</tbody>
</table>

#### SA Transform

<table>
<thead>
<tr>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

#### Proposal #2

<table>
<thead>
<tr>
<th>Proposal</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Type</td>
<td>1 (ENC)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>

#### Proposal #2

<table>
<thead>
<tr>
<th>Proposal</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>3 (more)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Type</td>
<td>3 (3DES)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform ID</td>
<td>According to above configuration</td>
</tr>
</tbody>
</table>
### Part A: Multiple Encryption Algorithms (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
6. Observe the messages transmitted on Link A.

### Part B: Multiple Pseudo Random Function (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
12. Observe the messages transmitted on Link A.

### Part C: Multiple Integrity Algorithms (BASIC)
13. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
14. Observe the messages transmitted on Link A.
15. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
16. Observe the messages transmitted on Link A.
17. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
18. Observe the messages transmitted on Link A.

### Part D: Multiple D-H Group (BASIC)
19. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
20. Observe the messages transmitted on Link A.
21. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
22. Observe the messages transmitted on Link A.
23. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT.
24. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Part B**

**Step 8: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 10: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 12: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Part C**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 16: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 18: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Part D**

**Step 20: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 22: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 24: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Possible Problems:**

- none
Test IKEv2.SGW.R.1.2.6.7: Changing RPFs when rekeying the IKE_SA

Purpose:

To verify an IKEv2 device properly handles CREATE_CHILD_SA to rekey IKE_SA.

References:

- [RFC 4306] - Sections 2.8
- [RFC 4718] - Sections 5.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. Configure the devices according to the Common Configuration except for *Italic* parameters.

<table>
<thead>
<tr>
<th>IKE_SA Rekeying Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption</td>
</tr>
<tr>
<td>Part A</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
</tbody>
</table>
Packet #3: CREATE_CHILD_SA request
Packet #3 is same as Common Packet #11 except SA Transform proposed in each test.

Part A:
SA Transform of Transform Type D-H is replaced by the following SA Transform.

<table>
<thead>
<tr>
<th>SA Transform</th>
<th>Next Payload</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Transform Length</th>
<th>Transform Type</th>
<th>Reserved</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 (last)</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>2 (PRF)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a SA payload. A proposal in the SA payload contains 1 (IKE) in the Protocol ID field, 8 in the SPI size field and the rekeyed IKE_SA Initiator’s SPI value.
6. Observe the messages transmitted on Link A.
7. TN1 transmits an INFORMATIONAL request with no payloads protected by the new IKE_SA and the Message ID field in the IKE header is zero.
8. Observe the messages transmitted on Link A.

Observables Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 14" as proposed algorithms. And the proposal in the SA payload includes 1 (IKE) in the Protocol ID field, 8 in the SPI size field and rekeyed IKE_SA Responder’s SPI value in the SPI field.

Step 8: Judgment #4
The NUT responds with an INFORMATIONAL response with no payloads protected by the new IKE_SA and the Message ID field in the IKE header is zero.

Possible Problems:
• none
Test IKEv2.SGW.R.1.2.6.8: D-H transform NONE when rekeying the IKE_SA

This test case was deleted at revision 1.1.0.
Test IKEv2.SGW.R.1.2.6.9: Rekeying Failure

Purpose:

To verify an IKEv2 device properly handles a CREATE_CHILD_SA request with an unacceptable SA payload.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

- Packet #1: See Common Packet #1
- Packet #2: See Common Packet #5
- Packet #3: See below

- Packet #3: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>N Payload</td>
<td>Same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Other fields are same as the Common Packet #15</td>
</tr>
<tr>
<td>SA Proposals</td>
<td>See below</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See below</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposal #</th>
<th>SA Proposal</th>
<th>Next Payload</th>
<th>0 (last)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal Length</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal #</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proposal ID</td>
<td>3 (ESP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI Size</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td># of Transforms</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPI</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>1 (ENCR)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>12 (AES_CBC)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>3 (more)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>2 (PRF)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>4 (AES128_CBC)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Next Payload</td>
<td>0 (last)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Length</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform Type</td>
<td>3 (INTEG)</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>SA Transform</td>
<td>Transform ID</td>
<td>5 (AES_CBC_96)</td>
</tr>
</tbody>
</table>

### Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request to rekey the established IKE_SA to the NUT. The CREATE_CHILD_SA request includes a SA payload with a proposal unaccepted by the NUT.
6. Observe the messages transmitted on Link A.

### Observable Results:

#### Part A

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits a CREATE_CHILD_SA response including a Notify payload of type NO_PROPOSAL_CHOSEN.

**Possible Problems:**

- None.
Group 2.7. Creating New CHILD_SA with the CREATE_CHILD_SA Exchange

Test IKEv2.SGW.R.1.2.7.1: Receipt of cryptographically protected message on the new SA

Purpose:

To verify an IKEv2 device properly recognizes the lifetime of CHILD_SAs.

References:

- [RFC 4306] - Sections 2.8

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:
<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #6</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #7</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #8</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #9</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #10</td>
<td>See below</td>
</tr>
<tr>
<td>Packet #11</td>
<td>See below</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

Packet #2: IKE_AUTH request

| IPv6 Header | Same as the Common Packet #5 |
| UDP Header   | Same as the Common Packet #5 |
| IKEv2 Header | Same as the Common Packet #5 |
| E Payload    | Same as the Common Packet #5 |
| IDi Payload  | Same as the Common Packet #5 |
| AUTH Payload | Same as the Common Packet #5 |
| N Payload    | Same as the Common Packet #5 |
| SA Payload   | Same as the Common Packet #5 |
| TSi Payload  | Other fields are same as the Common Packet #5 |
| TSr Payload  | Other fields are same as the Common Packet #5 |
| Traffic Selectors | See below |

<table>
<thead>
<tr>
<th>TSi Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH2's Global Address on Link B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH2's Global Address on Link B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSr Payload</th>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Starting Address</td>
<td>TH1's Global Address on Link Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ending Address</td>
<td>TH1's Global Address on Link Y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Packet #5: Echo Request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TN1's Global Address on Link X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td>NUT's Global Address on Link A</td>
<td></td>
</tr>
<tr>
<td>ESP</td>
<td>Security Parameter Index</td>
<td>CHILD_SA's SPI value used by this message</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet's Sequence Number by one.</td>
<td></td>
</tr>
<tr>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
<td></td>
</tr>
<tr>
<td>Pad Length</td>
<td>The length of the Padding field</td>
<td></td>
</tr>
<tr>
<td>Next Header</td>
<td>41 (IPv6)</td>
<td></td>
</tr>
<tr>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
<td></td>
</tr>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
<td>TH3's Global Address</td>
</tr>
<tr>
<td>Destination Address</td>
<td>TH1's Global Address</td>
<td></td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>128</td>
</tr>
<tr>
<td>Code</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Identifier</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Sequence Number</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Payload Data</td>
<td>0x0000000000000000</td>
<td></td>
</tr>
</tbody>
</table>

Packet #6: Echo Request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>TH1's Global Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td>TH3's Global Address</td>
<td></td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>128</td>
</tr>
<tr>
<td>Code</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Identifier</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Sequence Number</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>Payload Data</td>
<td>0x0000000000000000</td>
<td></td>
</tr>
</tbody>
</table>
Packet #7: CREATE_CHILD_SA request

<table>
<thead>
<tr>
<th>Packet</th>
<th>Payload Type</th>
<th>Traffic Selectors</th>
<th>TS Type</th>
<th>TSi Payload</th>
<th>TSr Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6</td>
<td>IPv6 Header</td>
<td>Same as the Common Packet #4</td>
<td>8 (IPv6_ADDR_RANGE)</td>
<td>IP Protocol ID 0 (any)</td>
<td>IP Protocol ID 0 (any)</td>
</tr>
<tr>
<td></td>
<td>UDP Header</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Selector Length 40</td>
<td>Selector Length 40</td>
</tr>
<tr>
<td></td>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Start Port 0</td>
<td>Start Port 0</td>
</tr>
<tr>
<td></td>
<td>E Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>End Port 65535</td>
<td>End Port 65535</td>
</tr>
<tr>
<td></td>
<td>IDi Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Starting Address TH3’s Global Address on Link B</td>
<td>Starting Address TH3’s Global Address on Link B</td>
</tr>
<tr>
<td></td>
<td>AUTH Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Ending Address TH3’s Global Address on Link B</td>
<td>Ending Address TH3’s Global Address on Link B</td>
</tr>
<tr>
<td></td>
<td>N Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See below</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Packet #10: Echo Request

<table>
<thead>
<tr>
<th>Packet</th>
<th>Payload Type</th>
<th>Traffic Selectors</th>
<th>TS Type</th>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
<th>ESP</th>
<th>Security Parameter Index</th>
<th>Sequence Number</th>
<th>Payload Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IPv6 Header</td>
<td>Same as the Common Packet #4</td>
<td>41 (IPv6)</td>
<td>IPv6 Header</td>
<td>TN1’s Global Address on Link X</td>
<td>NUT’s Global Address on Link A</td>
<td>ESP</td>
<td>CHILD_SA’s SPI value used by this message</td>
<td>0x0000000000000000</td>
<td>0x0000000000000000</td>
</tr>
<tr>
<td></td>
<td>UDP Header</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Security Parameter Index</td>
<td>CHILD_SA’s SPI value used by this message</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Sequence Number</td>
<td>The value incremented the previous encrypted packet’s Sequence Number by one.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>E Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Payload Data</td>
<td>Subsequent data encrypted by underlying encryption algorithm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IDi Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Padding</td>
<td>Any value which to be a multiple of the encryption block size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AUTH Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Pad Length</td>
<td>The length of the Padding field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Next Header</td>
<td>41 (IPv6)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA Payload</td>
<td>Same as the Common Packet #4</td>
<td></td>
<td>Integrity Check Value</td>
<td>The checksum must be valid by calculation according to the manner described in RFC.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TSi Payload</td>
<td>Other fields are same as the Common Packet #4</td>
<td></td>
<td>IPv6 Header</td>
<td>Source Address</td>
<td>TH3’s Global Address</td>
<td>Destination Address</td>
<td>TH1’s Global Address</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TSr Payload</td>
<td>Other fields are same as the Common Packet #4</td>
<td></td>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Traffic Selectors</td>
<td>See below</td>
<td></td>
<td>Code</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Identifier</td>
<td>any</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sequence Number</td>
<td>any</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Payload Data</td>
<td>0x0000000000000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Packet #11: Echo Reply

<table>
<thead>
<tr>
<th>Packet</th>
<th>Payload Type</th>
<th>Traffic Selectors</th>
<th>TS Type</th>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Destination Address</th>
<th>ICMPv6 Header</th>
<th>Type</th>
<th>Code</th>
<th>Identifier</th>
<th>Sequence Number</th>
<th>Payload Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IPv6 Header</td>
<td>Same as the Common Packet #4</td>
<td>129</td>
<td>IPv6 Header</td>
<td>TH1’s Global Address</td>
<td>TH3’s Global Address</td>
<td>ICMPv6 Header</td>
<td>Type</td>
<td>129</td>
<td>0</td>
<td>any</td>
<td>0x0000000000000000</td>
</tr>
</tbody>
</table>
Part A: (ADVANCED)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link B.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link B.
5. TH2 transmits an Echo Request packet to TH1.
6. Observe the messages transmitted on Link A.
7. TH1 transmits an Echo Reply packet to TH2.
8. Observe the messages transmitted on Link B.
9. TH3 transmits an Echo Request packet to TH1.
10. Observe the messages transmitted on Link A.
11. TH1 transmits an Echo Request packet to TH3.
12. Observe the messages transmitted on Link B.
13. TN1 starts to negotiate new CHILD_SA with the NUT by sending CREATE_CHILD_SA request.
14. Observe the messages transmitted on Link B.
15. TH2 transmits an Echo Request packet to TH1.
16. Observe the messages transmitted on Link A.
17. TH1 transmits an Echo Reply packet to TH2.
18. Observe the messages transmitted on Link B.
19. TH3 transmits an Echo Request packet to TH1.
20. Observe the messages transmitted on Link A.
21. TH1 transmits an Echo Reply packet to TH3.
22. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

Step 10: Judgment #5
The NUT never forwards an Echo Request.

Step 12: Judgment #6
The NUT never forwards an Echo Request with IPsec ESP using the first negotiated algorithms.
**Step 14: Judgment #7**
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 16: Judgment #8**
The NUT forwards an Echo Request.

**Step 18: Judgment #9**
The NUT forwards an Echo Reply with IPsec ESP using the first negotiated algorithms.

**Step 20: Judgment #10**
The NUT forwards an Echo Request.

**Step 22: Judgment #11**
The NUT forwards an Echo Reply with IPsec ESP using the second negotiated algorithms.

**Possible Problems:**

- None
Group 2.8. Error Handling

Test IKEv2.SGW.R.1.2.8.1: AUTHENTICATION_FAILED

This test case was deleted at revision 1.1.0.
Group 2.9. Non zero RESERVED fields

Test IKEv2.SGW.R.1.2.9.1: Non zero RESERVED fields in CREATE_CHILD_SA request

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #15</td>
</tr>
</tbody>
</table>

All RESERVED fields are set to one.

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits a CREATE_CHILD_SA request including a Notify Payload of type REKEY_SA and rekeyed CHILD_SA’s SPI value in the SPI field to the NUT. All RESERVED fields are set to one.
6. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits a CREATE_CHILD_SA response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:

• None.
Group 3. The INFORMATIONAL Exchange

Group 3.1. Header and Payload Formats

**Test IKEv2.SGW.R.1.3.1.1: Sending INFORMATIONAL response**

**Purpose:**

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

**References:**

- [RFC 4306] - Sections 1.1.2 and 1.4

**Test Setup:**

1. Network Topology
   - Connect the devices according to the Common Topology.
2. Configuration
   - In each part, configure the devices according to the Common Configuration.
3. Pre-Sequence and Cleanup Sequence
   - IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

**Part A: IKE Header Format (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT_SA response from the NUT, TN1 transmits an
IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an INFORMATIONAL request with no payloads to the NUT.
6. Observe the messages transmitted on Link A.

Part B: Encrypted Payload Format (BASIC)
7. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
8. Observe the messages transmitted on Link A.
9. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
10. Observe the messages transmitted on Link A.
11. After reception of IKE_AUTH response from the NUT, TN1 transmits an INFORMATIONAL request with no payloads to the NUT.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 7: Judgment #3
The NUT transmits an INFORMATIONAL response including properly formatted IKE Header containing following values:

- An IKE_SA Initiator’s SPI field set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field set to Encrypted Payload (46).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to INFORMATIONAL (37).
- A Flags field set to (00000100)\_2 = (4)\_10.
- A Message ID field set to the same value as corresponding IKEv2 request message’s Message ID.
- A Length field set to the length of the message (header + payloads) in octets.

**Part B**

**Step 9: Judgment #1**
The NUT transmits an IKE\_SA\_INIT response including "ENCR\_3DES", "PRF\_HMAC\_SHA1", "AUTH\_HMAC\_SHA1\_96" and "D\-H Group 2" as proposed algorithms.

**Step 11: Judgment #2**
The NUT transmits an IKE\_AUTH response including "ENCR\_3DES", "AUTH\_HMAC\_SHA1\_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 14: Judgment #3**
The NUT transmits an INFORMATIONAL response including properly formatted Encrypted Payload containing following values:

- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Checksum Data.
- An Initialization Vector field set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR\_3DES case.
- An Encrypted IKE Payloads field set to subsequent payloads encrypted by ENCR\_3DES.
- A Padding field set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR\_3DES case.
- A Pad Length field set to the length of the Padding field.
- An Integrity Checksum Data set to the cryptographic checksum of the entire

**Figure 172 Encrypted payload**
message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

**Possible Problems:***

- None.
Group 3.2. Use of Retransmission Timers

Test IKEv2.SGW.R.1.3.2.1: Receipt of retransmitted INFORMATIONAL request

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 1.1.2, 1.4 and 2.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an INFORMATIONAL request with no payloads.
6. Observe the messages transmitted on Link A.
7. Observe the messages transmitted on Link A.
8. TN1 transmits an INFORMATIONAL request with no payloads. The Message ID is the same as Step 5.
9. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

Step 6: Judgment #3
The NUT transmits an INFORMATIONAL response followed by an Encrypted payload with no payloads contained in it.

Step 7: Judgment #4
The NUT transmits an INFORMATIONAL response followed by an Encrypted payload with no payloads contained in it.

Step 9: Judgment #5
The NUT transmits an INFORMATIONAL response followed by an Encrypted payload with no payloads contained in it.

Possible Problems:

- None
Group 3.3. Non zero RESERVED fields

Test IKEv2.SGW.R.1.3.3.1: Non RESERVED fields in INFORMATIONAL request

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration. In addition, set IKE_SA Lifetime to 300 seconds and set CHILD_SA Lifetime to 30 seconds.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #17</td>
</tr>
</tbody>
</table>

All RESERVED fields are set to one.

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_AUTH response from the NUT, TN1 transmits an IKE_AUTH
request to the NUT.
4. Observe the messages transmitted on Link A.
5. After reception of IKE_AUTH response from the NUT, TN1 transmits an
   INFORMATIONAL request with no payloads. All RESERVED fields in the message are set
to one.
6. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES",
"PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as accepted
algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES",
"AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as accepted algorithms.

**Step 6: Judgment #3**
The NUT transmits an INFOMATIONAL Response followed by an Encrypted payload with
no payloads contained in it.

**Possible Problems:**

- None
Group 4. RFC 5996

Group 4.1. Rekeying IKE SAs Using a CREATE_HLD_SA Exchange


Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 5996] - Section 2.18

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 1017 IPv6 Ready Logo Program IKEv2
IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Packet #3</th>
<th>See Common Packet #21</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See Common Packet #11</td>
</tr>
</tbody>
</table>

**Part A: (BASIC)**

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. TN1 transmits a CREATE_CHILD_SA request including a SA payload. A proposal in the SA payload contains 1 (IKE) in the Protocol ID field, 8 in the SPI size field and the rekeyed IKE_SA’s initiator’s SPI value. The proposal has the value "NONE" for the Diffie-Hellman transform.
10. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 6: Judgment #3**
The NUT forwards an Echo Request.

**Step 8: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Step 10: Judgment #4**
The NUT transmits a CREATE_CHILD_SA response with Notify payload of type NO_PROPOSAL_CHOSEN.

**Possible Problems:**

- None
[SGW.R.P86.L4030.ADD.1] Test IKEv2.EN.R.1.4.1.2.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can handle a proposal that contains a Transform Type it does not understand

References:

- [RFC 5996] - Section 3.3.6

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IKE_SA_INIT request has 2 SA Proposals.</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request. The request has 2 proposals. One proposal has 5 Transforms which are "a Transform Type (240)", "ENC_R_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", and "D-H Group 2". Another proposal has 4 Transforms which are "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", and "D-H Group 2".
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Possible Problems:

- None
[SGW.R.P86.L4030.ADD.2] Test IKEv2.EN.R.1.4.1.3.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can handle a proposal that is missing a mandatory Transform Type

References:

- [RFC 5996] - Section 3.3.6

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

```
Packet #1       See Common Packet #1
IKE_SA_INIT request has 2 SA Proposals.
```

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request. The request has 2 proposals. One proposal has 3 Transforms which "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", and "D-H Group 2". Another proposal has 4 Transforms which are "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", and "D-H Group 2".
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.
Possible Problems:

- None
[SGW.R.P86.L4034.ADD.1] Test IKEv2.EN.R.1.4.1.4.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can handle a transform that it does not understand

References:

- [RFC 5996] - Section 3.3.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKE_SA_INIT request has 5 SA Transforms.</td>
<td></td>
</tr>
</tbody>
</table>

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request. The request has 5 Transforms which are "1 (ENCR) as Transform Type and 1023 as Transform ID", "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2".
2. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1

The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Possible Problems:

- None
Test IKEv2.EN.R.1.4.1.5.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can handle a Transform Attribute it does not understand

References:

- [RFC 5996] - Section 3.3.5

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>IKE_SA_INIT request has 5 SA Transforms.</th>
</tr>
</thead>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request. The request has 5 Transforms which are "ENCR_3DES with Transform Attribute of type KeyLength and value 192", "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96", "D-H Group 2"
2. Observe the messages transmitted on Link A.

Observable Results:

Part A Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Possible Problems:

- None
Purpose:

To verify an IKEv2 device can notify AUTHENTICATION FAILED

References:

- [RFC 5996] - Section 2.21.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>Authentication Data is 0x0123456789abcdef0123456789abcdef01234567.</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT. The response includes invalid Authentication Data 0x0123456789abcdef0123456789abcdef01234567.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including Notify payload of type AUTHENTICATION_FAILED.

**Possible Problems:**

- None
[SGW.R.P69.L3234.ADD] Test IKEv2.EN.R.1.4.1.7.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can process CHILD_CHILD_SA request to close a Child SA that it is currently rekeying.

References:

- [RFC 5996] - Section 2.25.1

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See below</td>
</tr>
</tbody>
</table>

IPv6 FORUM TECHNICAL DOCUMENT 1028 IPv6 Ready Logo Program IKEv2
Packet #5: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Delete Payload</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
</tr>
<tr>
<td></td>
<td># of SPIs</td>
</tr>
<tr>
<td></td>
<td>Security Parameter Index(es) (SPI)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. Repeat Steps 5 and 6 until lifetime of SA is expired for 30 seconds.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request. The request includes a Delete payload with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA request to rekey a Child SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers".

Numbers” as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

**Step 12: Judgment #6**
The NUT transmits an INFORMATIONAL response. The response includes a Delete payload with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.

**Possible Problems:**

- None
Purpose:

To verify an IKEv2 device can process INFORMATIONAL request to close IKE SA that it is currently rekeying.

References:

- [RFC 5996] - Section 2.25.1

Test Setup:

- Network Topology
  - Connect the devices according to the Common Topology.
- Configuration
  - In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
</tbody>
</table>
Packet #4: See Common Packet #25
Packet #5: See Common below

- Packet #5: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>42 (Delete)</td>
</tr>
<tr>
<td>Delete Payload</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>16</td>
</tr>
<tr>
<td>Protocol ID</td>
<td>1 (IKE_SA)</td>
</tr>
<tr>
<td>SPI Size</td>
<td>0</td>
</tr>
<tr>
<td># of SPIs</td>
<td>0</td>
</tr>
<tr>
<td>Security Parameter Index(es) (SPI)</td>
<td>empty</td>
</tr>
</tbody>
</table>

Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. Repeat Steps 5 and 6 until lifetime of SA is expired for 30 seconds.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request. The request includes a Delete payload with 1 (IKE) as Protocol ID, zero as SPI Size and no SPI value.
12. Observe the messages transmitted on Link A.
13. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA request to rekey IKE SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 12: Judgment #6**
The NUT transmits an INFORMATIONAL response with no payloads.

**Step 13: Judgment #7**
The NUT does not retransmit a CREATE_CHILD_SA request to rekey a Child SA.

**Possible Problems:**

- None
[SGW.R.P69.L3258.ADD] Test IKEv2.EN.R.1.4.1.9.XXX: Sending INFORMATIONAL Exchange

Purpose:

To verify an IKEv2 device can process CREATE_CHILD_SA request to rekey a Child SA when it is currently rekeying the IKE SA.

References:

- [RFC 5996] - Section 2.5.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
</tbody>
</table>
Part A: (BASIC)
1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. Repeat Steps 5 and 6 until lifetime of SA is expired for 30 seconds.
10. Observe the messages transmitted on Link A.
11. TN1 transmits a CREATE_CHILD_SA request to rekey a Child SA. The request includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT transmits a CREATE_CHILD_SA request to rekey a Child SA. The message includes "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. And the CREATE_CHILD_SA request includes a Notify payload of type REKEY_SA containing rekeyed CHILD_SA’s SPI value in the SPI field.

Step 12: Judgment #6
The NUT transmits a CREATE_CHILD_SA response. The response includes a Notify payload of type TEMPORARY_FAILURE.

Possible Problems:

- None
**Purpose:**

To verify an IKEv2 device can process CREATE_CHILD_SA request to delete a Child SA when it is currently rekeying the IKE SA

**References:**

- [RFC 5996] - Section 2.5.2

**Test Setup:**

- **Network Topology**
  - Connect the devices according to the Common Topology.
- **Configuration**
  - In each part, configure the devices according to the Common Configuration.
- **Pre-Sequence and Cleanup Sequence**
  - IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
<tr>
<td>Packet #3</td>
<td>See Common Packet #21</td>
</tr>
<tr>
<td>Packet #4</td>
<td>See Common Packet #25</td>
</tr>
<tr>
<td>Packet #5</td>
<td>See below</td>
</tr>
</tbody>
</table>

---

IPv6 FORUM TECHNICAL DOCUMENT 1036 IPv6 Ready Logo Program IKEv2
IPv6 FORUM TECHNICAL DOCUMENT

Packet #5: INFORMATIONAL request

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as the Common Packet #17</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as the Common Packet #17</td>
</tr>
<tr>
<td>E Payload</td>
<td>Other fields are same as the Common Packet #17</td>
</tr>
<tr>
<td></td>
<td>Next Payload</td>
</tr>
<tr>
<td>Delete Payload</td>
<td>Next Payload</td>
</tr>
<tr>
<td></td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td></td>
<td>Payload Length</td>
</tr>
<tr>
<td></td>
<td>Protocol ID</td>
</tr>
<tr>
<td></td>
<td>SPI Size</td>
</tr>
<tr>
<td></td>
<td># of SPIs</td>
</tr>
<tr>
<td></td>
<td>Security Parameter Index(es) (SPI)</td>
</tr>
</tbody>
</table>

Part A: (BASIC)

1. TN1 starts to negotiate with NUT by sending IKE_SA_INIT request.
2. Observe the messages transmitted on Link A.
3. After reception of IKE_SA_INIT response from the NUT, TN1 transmits an IKE_AUTH request to the NUT.
4. Observe the messages transmitted on Link A.
5. TH2 transmits an Echo Request to TH1.
6. Observe the messages transmitted on Link B.
7. TH1 transmits an Echo Reply to TH2.
8. Observe the messages transmitted on Link A.
9. Repeat Steps 5 and 6 until lifetime of SA is expired for 30 seconds.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an INFORMATIONAL request. The request includes a Delete payload with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.
12. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
TN1 transmits an INFORMATIONAL request. The request includes a Delete payload with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.
Step 12: Judgment #6
The NUT transmits an INFORMATIONAL response with 3 (ESP) as Protocol ID, 4 as SPI Size and SPI value to delete Child SA.

Possible Problems:

- None
Section 2.2.2. Endpoint to Security Gateway Tunnel
Group 1. The Initial Exchanges
Group 1.1. Header and Payload Formats

Test IKEv2.SGW.R.2.1.1.1: Sending IKE_AUTH response

Purpose:
To verify an IKEv2 device transmits IKE_AUTH request using properly Header and Payloads format

References:
- [RFC 4306] - Sections 1.2, 2.15, 3.1, 3.2, 3.3, 3.5, 3.8, 3.10, 3.13 and 3.14

Test Setup:
- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
</tbody>
</table>

Part A: IKE Header Format (BASIC)
1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT request to NUT.
4. Observe the messages transmitted on Link A.

Part B: Encrypted Payload Format (BASIC)
5. TN1 transmits an IKE_SA_INIT request to NUT.
6. Observe the messages transmitted on Link A.
7. TN1 transmits an IKE_SA_INIT request to NUT.
8. Observe the messages transmitted on Link A.

**Part C: IDr Payload Format (BASIC)**
9. TN1 transmits an IKE_SA_INIT request to NUT.
10. Observe the messages transmitted on Link A.
11. TN1 transmits an IKE_SA_INIT request to NUT.
12. Observe the messages transmitted on Link A.

**Part D: AUTH Payload Format (BASIC)**
13. TN1 transmits an IKE_SA_INIT request to NUT.
14. Observe the messages transmitted on Link A.
15. TN1 transmits an IKE_SA_INIT request to NUT.
16. Observe the messages transmitted on Link A.

**Part E: SA Payload Format (BASIC)**
17. TN1 transmits an IKE_SA_INIT request to NUT.
18. Observe the messages transmitted on Link A.
19. TN1 transmits an IKE_SA_INIT request to NUT.
20. Observe the messages transmitted on Link A.

**Part F: TSi Payload Format (BASIC)**
21. TN1 transmits an IKE_SA_INIT request to NUT.
22. Observe the messages transmitted on Link A.
23. TN1 transmits an IKE_SA_INIT request to NUT.
24. Observe the messages transmitted on Link A.

**Part G: TSr Payload Format (BASIC)**
25. TN1 transmits an IKE_SA_INIT request to NUT.
26. Observe the messages transmitted on Link A.
27. TN1 transmits an IKE_SA_INIT request to NUT.
28. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted IKE Header containing following values:
Figure 173 Header format

- An IKE_SA Initiator’s SPI field set to same as the IKE_SA_INIT request’s IKE_SA Initiator’s SPI field value.
- An IKE_SA Responder’s SPI field set to same as the IKE_SA_INIT response’s IKE_SA Responder’s SPI field value.
- A Next Payload field set to Encrypted Payload (46).
- A Major Version field set to 2.
- A Minor Version field set to zero.
- An Exchange Type field set to IKE_AUTH (35).
- A Flags field set to (00010000)2 = (16)10.
- A Message ID field set to 1.
- A Length field set to the length of the message (header + payloads) in octets.

Part B

Step 6: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 8: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted Encrypted Payload containing following values:

Figure 174 Encrypted payload
• A Next Payload field set to IDr Payload (36).
• A Critical field set to zero.
• A RESERVED field set to zero.
• A Payload Length field set to length in octets of the header, IV, Encrypted IKE Payloads, Padding, Pad Length, and Integrity Check sum Data.
• An Initialization Vector field set to a randomly chosen value whose length is equal to the block length of the underlying encryption algorithm. It is 64 bits length in ENCR_3DES case.
• An Encrypted IKE Payloads field set to subsequent payloads encrypted by ENCR_3DES.
• A Padding field set to any value which to be a multiple of the encryption block size. It is 64 bits length in ENCR_3DES case.
• A Pad Length field set to the length of the Padding field.
• An Integrity Checksum Data set to the cryptographic checksum of the entire message. It is 96 bits length in AUTH_HMAC_SHA1_96 case. The checksum must be valid by calculation according to the manner described in RFC.

**Part C**

**Step 10: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 12: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted ID Payload containing following values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Payload</td>
<td>AUTH Payload (39).</td>
</tr>
<tr>
<td>Critical</td>
<td>set to zero.</td>
</tr>
<tr>
<td>RESERVED</td>
<td>set to zero.</td>
</tr>
<tr>
<td>Payload Length</td>
<td>set to length of the current payload. It is 24 bytes for ID_IPV6_ADDR.</td>
</tr>
<tr>
<td>ID Type</td>
<td>set to ID_IPV6_ADDR (5).</td>
</tr>
<tr>
<td>RESERVED</td>
<td>set to zero.</td>
</tr>
<tr>
<td>Identification Data</td>
<td>set to the NUT address.</td>
</tr>
</tbody>
</table>

**Figure 175 ID Payload format**

**Part D**

**Step 14: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 16: Judgment #2**

The NUT transmits an IKE_AUTH response including properly formatted AUTH Payload containing following values:

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload. It is 28 bytes for PRF_HMAC_SHA1.
- An Auth Method field set to Shared Key Message Integrity Code (2).
- A RESERVED field set to zero.
- An Authentication Data field set to correct authentication value according to the manner described in RFC. It is 160 bytes length in PRF_HMAC_SHA1 case.

**Part E**

**Step 18: Judgment #1**

The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 20: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted SA Payload containing following values (refer following figures):

- A Next Payload field set to TSi Payload (44).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.

A Proposals field set to following.
Proposal #1

- A 0 or 2 field set to zero (last).
- A RESERVD field set to zero.
- A Proposal Length field set to length of this proposal, including all transforms and attributes. It is 36 bytes according to Common Configuration.
- A Proposal # field set to 1.
- A Protocol ID field set to ESP (3).
- A SPI Size field set to 4.
- A # of Transforms field set to 3.
- A SPI field set to the sending entity’s SPI (4 octets value)

Transform field set to following (There are 3 Transform Structures).
Header and Attribute. It is 8 bytes for AUTH_HMAC_SHA1.
- A Transform Type field set to INTEG (3).
- A RESERVED field set to zero.
- A Transform ID set to AUTH_HMAC_SHA1 (2).

Transform #3
- A 0 or 3 field set to zero if this structure is the last transform, otherwise set to 3.
- A RESERVED field set to zero.
- A Transform Length set to length of the Transform Substructure including Header and Attribute. It is 8 bytes for ESN.
- A Transform Type field set to ESN (5).
- A RESERVED field set to zero.
- A Transform ID set to No Extended Sequence Numbers (0).

Part F

Step 22: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 24: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted TSi Payload containing following values:

![Figure 181 TSi Payload format](image)

- A Next Payload field set to TSr Payload (45).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to 1.
- A RESERVED field set to zero.

Traffic Selectors field set to following.
Figure 182 Traffic Selector

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to TN1 address.
- A Ending Address field set to greater than or equal to TN1 address.

Part G

Step 26: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 28: Judgment #2
The NUT transmits an IKE_AUTH response including properly formatted TSr Payload containing following values:

Figure 183 TSr Payload format

- A Next Payload field set to zero.
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A Number of TSs field set to 1.
- A RESERVED field set to zero.
Traffic Selectors field set to following.

![Figure 184 Traffic Selector](image)

- A TS Type set to TS_IPV6_ADDR_RANGE (8).
- An IP Protocol ID field set to zero.
- A Selector Length field set to length of this Traffic Selector Substructure including the header. It is 40 bytes for TS_IPV6_ADDR_RANGE.
- A Start Port field set to zero.
- An End Port field set to 65535.
- A Starting Address field set to less than or equal to Prefix B.
- An Ending Address field set to less than or equal to Prefix B.

Possible Problems:

- IKE_AUTH response has following packet format. It may have additional payloads described below. Additional payloads can be ignored by this test. The order of payload may be different from this sample.

![Figure 184 Traffic Selector](image)

- Each of transforms can be located in the any order.
Test IKEv2.SGW.R.2.1.1.2: Use of CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 1.2

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #2</th>
<th>Packet #2</th>
<th>See Common Packet #6</th>
</tr>
</thead>
</table>

Part A (BASIC)

1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT response to NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to TH1.
6. Observe the messages transmitted on Link A.
7. TH1 transmits an Echo Reply to TN1.
8. Observe the messages transmitted on Link B.

Observable Results:
Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request.

Step 8 Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

- None.
Group 1.2. Requesting an Internal Address on a Remote Network

Test IKEv2.SGW.R.2.1.2.1: Receipt of CFG_REQUEST

Purpose:
To verify an IKEv2 device transmits IKE_AUTH request using properly eader and Configuration Payload format

References:
- [RFC 4306] - Sections 3.15

Test Setup:
- Network Topology
  Connect the devices according to the following topology.

  ![Network Topology Diagram]

  - Prefix A = 2001:0dB8:0001:0001::/64
  - Prefix B = 2001:0dB8:0001:0002::/64
  - Prefix X = 2001:0dB8:000f:0001::/64

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic.
selector. Configure NUT to transmit CFG_REPLY for INTERNAL_IP6_ADDRESS. Its IPv6 address is Prefix B::1/128. The traffic selector must be configured by the following table. NUT must narrow Traffic Selector to the following address range.

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Source</th>
<th>Next Layer</th>
<th>Protocol</th>
<th>Port</th>
<th>Range</th>
<th>Destination</th>
<th>Next Layer</th>
<th>Protocol</th>
<th>Port</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbound</td>
<td>TN1</td>
<td>ANY</td>
<td>ANY</td>
<td>Link B</td>
<td>ANY</td>
<td>INTERNAL_IP6_ADDRESS</td>
<td>ANY</td>
<td>ANY</td>
<td>ANY</td>
<td></td>
</tr>
<tr>
<td>Outbound</td>
<td>Link B</td>
<td>ANY</td>
<td>ANY</td>
<td>TN1</td>
<td>ANY</td>
<td>(internal address)</td>
<td>ANY</td>
<td>ANY</td>
<td>ANY</td>
<td></td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

**Procedure:**

- Packet #1: See Common Packet #1
- Packet #2: See below

**Packet #2: IKE_AUTH request packet**

<table>
<thead>
<tr>
<th>Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Next Payload 47 (CP)</td>
</tr>
<tr>
<td>CP Payload</td>
<td>Next Payload 33 (SA)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Payload Length</td>
<td>12</td>
</tr>
<tr>
<td>CFG Type</td>
<td>1 (CFG_REQUEST)</td>
</tr>
<tr>
<td>RESERVED</td>
<td>0</td>
</tr>
<tr>
<td>Configuration Attributes</td>
<td>See below</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as Common Packet #5</td>
</tr>
<tr>
<td>Traffic Selectors</td>
<td>See below</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as Common Packet #5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Configuration Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
<tr>
<td>Attribute Type</td>
<td>INTERNAL_IP6_ADDRESS</td>
</tr>
<tr>
<td>Length</td>
<td>0</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

IPv6 Ready Logo Program IKEv2

IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Protocol ID</td>
<td>0 (any)</td>
<td></td>
</tr>
<tr>
<td>Selector Length</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
<td></td>
</tr>
</tbody>
</table>

**Part A: (ADVANCED)**

1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT request to NUT.
4. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including properly formatted AUTH Payload containing following values:

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A CFG Type field set to CFG_REPLY (2).
- A RESERVED field set to zero.

A Configuration Attributes field set to following.

**Figure 185 Configuration Payload format**

- A Next Payload field set to SA Payload (33).
- A Critical field set to zero.
- A RESERVED field set to zero.
- A Payload Length field set to length of the current payload.
- A CFG Type field set to CFG_REPLY (2).
- A RESERVED field set to zero.

A Configuration Attributes field set to following.
### Figure 186 Configuration Attributes format

**Configuration Attribute #1**
- Reserved field is set to zero.
- Attribute Type field is set to INTERNAL_IP6_ADDRESS (8).
- Length field is set to 17.
- Value field is set to Prefix B::1 as IPv6 address and 128 as prefix-length.

**Possible Problems:**
- None.
Test IKEv2.SGW.R.2.1.2.2: Use of CHILD_SA

Purpose:

To verify an IKEv2 device properly handles the Initial Exchanges using Pre-shared key

References:

- [RFC 4306] - Sections 2.19 and 3.15

Test Setup:

- Network Topology
  Connect the devices according to the following topology.

<table>
<thead>
<tr>
<th>Prefix B::f</th>
<th>Link B (Prefix B, MTU=1500)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Prefix B::any_interface_ID</td>
</tr>
<tr>
<td></td>
<td>NUT (SGW)</td>
</tr>
<tr>
<td></td>
<td>Prefix A::any_interface_ID</td>
</tr>
<tr>
<td></td>
<td>Link A (Prefix A, MTU=1500)</td>
</tr>
<tr>
<td></td>
<td>Prefix X::1 (External Address)</td>
</tr>
<tr>
<td></td>
<td>Prefix B::1 (Internal Address) (assigned by CP)</td>
</tr>
</tbody>
</table>

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REPLY for INTERNAL_IP6_ADDRESS. Its IPv6 address is Prefix B::1/128. The traffic selector must be configured by the following table. NUT must narrow Traffic Selector to the following address table.
Pre-Sequence and Cleanup Sequence
IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #2: IKE_AUTH request packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #5</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>IDI Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>CP Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>SA Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>TSi Payload</td>
<td>Other fields are same as Common Packet #5</td>
</tr>
<tr>
<td>TSr Payload</td>
<td>Same as Common Packet #5</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

<table>
<thead>
<tr>
<th>Configuration Attributes</th>
<th>Reserved</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Type</td>
<td>INTERNAL_IPV6_ADDRESS</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>TS Type</th>
<th>8 (IPV6_ADDR_RANGE)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IP Protocol ID</td>
<td>0 (any)</td>
</tr>
<tr>
<td></td>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td></td>
<td>Starting Address</td>
<td>::</td>
</tr>
</tbody>
</table>

- Packet #3: Echo Request packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #22</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESP</td>
<td>Same as Common Packet #22</td>
</tr>
<tr>
<td>IPv6 Header</td>
<td>Source Address</td>
</tr>
<tr>
<td></td>
<td>Destination Address</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Same as Common Packet #22</td>
</tr>
</tbody>
</table>

- Packet #4: Echo Reply packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Source Address</th>
<th>Prefix B::f</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Destination Address</td>
<td>Prefix B::1</td>
</tr>
<tr>
<td>ICMPv6 Header</td>
<td>Same as Common Packet #26</td>
<td></td>
</tr>
</tbody>
</table>

**Part A (ADVANCED)**

1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to TH1.
6. Observe the messages transmitted on Link A.
7. TH1 transmits an Echo Reply to TN1.
8. Observe the messages transmitted on Link B.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

**Step 6: Judgment #3**
The NUT forwards an Echo Request to the TH1.

**Step 8: Judgment #4**
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

**Possible Problems:**
• Because the destination address of Echo Request is the TN itself, TN may respond to Echo Request automatically. In that case, TN1 can send Echo Reply to TH1 instead of sending Echo Request.
Test IKEv2.SGW.R.2.1.2.3: Non zero RESERVED fields in Configuration Payload

Purpose:

To verify an IKEv2 device ignores the content of RESERVED filed in IKE messages.

References:

- [RFC 4306] - Sections 2.5

Test Setup:

- Network Topology
  Connect the devices according to the following topology.

  ![Network Topology Diagram]

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REPLY for INTERNAL_IP6_ADDRESS. Its IPv6 address is Prefix B::1/128. The traffic selector must be configured by the following table. NUT must narrow Traffic Selector to the following address table.
Pre-Sequence and Cleanup Sequence
IKEv2 on the NUT is disabled after each part.

Procedure:

Packet #1: See Common Packet #5
Packet #2: See below

Packet #2: IKE_AUTH request packet

IPv6 Header: Same as Common Packet #5
UDP Header: Same as Common Packet #5
IKEv2 Header: Same as Common Packet #5
E Payload: Same as Common Packet #5
IDi Payload: Same as Common Packet #5
AUTH Payload: Next Payload 47 (CP)
Other fields are same as Common Packet #5

CP Payload: Next Payload 33 (SA)
Critical: 0
Reserved: 1
Payload Length: 12
CFG Type: 1 (CFG_REQUEST)
RESERVED: 1
Configuration Attributes: See below

SA Payload: Same as Common Packet #5
TSi Payload: Other fields are same as Common Packet #5
Traffic Selectors: See below
TSr Payload: Same as Common Packet #5

Configuration Attributes: Reserved: 1
Attribute Type: INTERNAL_IP6_ADDRESS
Length: 0

Traffic Selector: TS Type: 8 (IPV6_ADDR_RANGE)
IPv6 FORUM TECHNICAL DOCUMENT 1062

IPv6 Ready Logo Program IKEv2

<table>
<thead>
<tr>
<th>IP Protocol ID</th>
<th>0 (any)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selector Length</td>
<td>40</td>
</tr>
<tr>
<td>Start Port</td>
<td>0</td>
</tr>
<tr>
<td>End Port</td>
<td>65535</td>
</tr>
<tr>
<td>Starting Address</td>
<td>::</td>
</tr>
</tbody>
</table>

Part A (ADVANCED)
1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT request to the NUT.
4. Observe the messages transmitted on Link A.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Possible Problems:
- None.
Test IKEv2.SGW.R.2.1.2.4: No Configuration payload

Purpose:

To verify an IKEv2 device properly handles the message which does not include Configuration payload, when the device expects Configuration payload.

References:

- [RFC 4306] - Sections 2.19 and 3.10.1

Test Setup:

- Network Topology
  Connect the devices according to the following topology.

  ![Diagram of network topology]

  Prefix A = 2001:0db8:0001:0001::/64
  Prefix B = 2001:0db8:0001:0002::/64
  Prefix X = 2001:0db8:000f:0001::/64

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REPLY for INTERNAL_IP6_ADDRESS. Its IPv6 address is Prefix B::1/128. The traffic selector must be configured by the following table. NUT must narrow Traffic Selector to the following address table.
### Traffic Selector

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address Range</th>
<th>Next Layer Protocol</th>
<th>Port Range</th>
<th>Address Range</th>
<th>Next Layer Protocol</th>
<th>Port Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>TN1 (internal address)</td>
<td>ANY</td>
<td>ANY</td>
<td>Link B</td>
<td>ANY</td>
<td>ANY</td>
</tr>
<tr>
<td>Link B</td>
<td>ANY</td>
<td>ANY</td>
<td>TN1 (internal address)</td>
<td>ANY</td>
<td>ANY</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  - IKEv2 on the NUT is disabled after each part.

**Procedure:**

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet #2</td>
<td>See Common Packet #5</td>
</tr>
</tbody>
</table>

**Part A (ADVANCED)**

1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT request to the NUT.
4. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
- The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
- The NUT transmits an IKE_AUTH response with a Notify payload of type FAILED_CP_REQUIRED.

**Possible Problems:**
- None.
Test IKEv2.SGW.R.2.1.2.5: Receipt of Multiple CFG_REQUEST

Purpose:

To verify an IKEv2 device properly handles multiple CFG_REQUEST.

References:

- [RFC 4306] - Sections 2.19 and 3.15

Test Setup:

- Network Topology
  Connect the devices according to the following topology.

```
Prefix B::2 (Internal Address) (assigned by CP)
Link B (Prefix B, MTU=1500)
Link A (Prefix A, MTU=1500)
Link X (Prefix X, MTU=1500)
Prefix B::f
Prefix A = 2001:0db8:0001:0001::/64
Prefix B = 2001:0db8:0001:0002::/64
Prefix X = 2001:0db8:000f:0001::/64
Prefix B::any_interface_ID
Prefix A::any_interface_ID
fe80::f
Prefix X::1 (External Address)
Prefix X::1 (External Address)
Prefix B::1 (Internal Address) (assigned by CP)
Prefix B::2 (Internal Address) (assigned by CP)
```

- TH1 (Host)
- NUT (SGW)
- TR1 (Router)
- TN1 (End-Node)

- Configuration
  In each part, configure NUT according to the Common Configuration except the traffic selector. Configure NUT to transmit CFG_REPLY for INTERNAL_IP6_ADDRESS. Its IPv6 address is Prefix B::1/128. The traffic selector must be configured by the following table. NUT must narrow Traffic Selector to the following address table.
**IPv6 FORUM TECHNICAL DOCUMENT**

<table>
<thead>
<tr>
<th>Traffic Selector</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Next Layer Protocol</td>
</tr>
<tr>
<td>Inbound</td>
<td>TN1 (internal address)</td>
</tr>
<tr>
<td>Outbound</td>
<td>Link B</td>
</tr>
</tbody>
</table>

- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

**Procedure:**

Packet #2: IKE_AUTH request packet

<table>
<thead>
<tr>
<th>IPv6 Header</th>
<th>Same as Common Packet #5</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP Header</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>IKEv2 Header</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>E Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>IDi Payload</td>
<td>Same as Common Packet #5</td>
</tr>
<tr>
<td>AUTH Payload</td>
<td>Next Payload 47 (CP)</td>
</tr>
<tr>
<td>Other fields are same as Common Packet #5</td>
<td></td>
</tr>
<tr>
<td>CP Payload</td>
<td>Next Payload 33 (SA)</td>
</tr>
<tr>
<td>Critical</td>
<td>0</td>
</tr>
<tr>
<td>Reserved</td>
<td>0</td>
</tr>
</tbody>
</table>
IPv6 FORUM TECHNICAL DOCUMENT

IPv6 Ready Logo Program IKEv2

Payload Length 16
CFG Type 1 (CFG_REQUEST)
RESERVED 0
Configuration Attributes See below
SA Payload Same as Common Packet #5
TSi Payload Other fields are same as Common Packet #5
Traffic Selectors See below
TSr Payload Same as Common Packet #5

Configuration Attributes

<table>
<thead>
<tr>
<th>Reserved</th>
<th>Attribute Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>INTERNAL_IP6_ADDRESS</td>
<td>0</td>
</tr>
</tbody>
</table>

Configuration Attributes

<table>
<thead>
<tr>
<th>Reserved</th>
<th>Attribute Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>INTERNAL_IP6_ADDRESS</td>
<td>0</td>
</tr>
</tbody>
</table>

Traffic Selector

<table>
<thead>
<tr>
<th>TS Type</th>
<th>IP Protocol ID</th>
<th>Selector Length</th>
<th>Start Port</th>
<th>End Port</th>
<th>Starting Address</th>
<th>Ending Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 (IPV6_ADDR_RANGE)</td>
<td>0 (any)</td>
<td>40</td>
<td>0</td>
<td>65535</td>
<td>::</td>
<td>ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff</td>
</tr>
</tbody>
</table>

Packet #3: Echo Request packet

IPv6 Header Same as Common Packet #22
ESP Same as Common Packet #22
IPv6 Header Source Address Prefix B::1
Destination Address Prefix B::f
ICMPv6 Header Same as Common Packet #22

Packet #4: Echo Reply packet

IPv6 Header Source Address Prefix B::f
Destination Address Prefix B::1
ICMPv6 Header Same as Common Packet #26

Packet #5: Echo Request packet

IPv6 Header Same as Common Packet #22
ESP Same as Common Packet #22
IPv6 Header Source Address Prefix B::2
Destination Address Prefix B::f
ICMPv6 Header Same as Common Packet #22

Packet #6: Echo Reply packet

IPv6 Header Source Address Prefix B::f
Destination Address Prefix B::2
ICMPv6 Header Same as Common Packet #26

Part A (ADVANCED)

1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT request to the NUT.
4. Observe the messages transmitted on Link A.
5. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to TH1.
6. Observe the messages transmitted on Link A.
7. TH1 transmits an Echo Reply to TN1.
8. Observe the messages transmitted on Link B.
9. TN1 transmits an Echo Request with IPsec ESP using corresponding algorithms to TH1.
10. Observe the messages transmitted on Link A.
11. TH1 transmits an Echo Reply to TN1.
12. Observe the messages transmitted on Link B.

Observable Results:

Part A

Step 2: Judgment #1
The NUT transmits an IKE_SA_INIT response including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

Step 4: Judgment #2
The NUT transmits an IKE_AUTH response including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms.

Step 6: Judgment #3
The NUT forwards an Echo Request to the TH1.

Step 8: Judgment #4
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Step 10: Judgment #5
The NUT forwards an Echo Request to the TH1.

Step 12: Judgment #6
The NUT forwards an Echo Reply with IPsec ESP using corresponding algorithms.

Possible Problems:

• Because the destination address of Echo Request is the TN itself, TN may respond to Echo Request automatically. In that case, TN1 can send Echo Reply to TH1 instead of sending Echo Request.
Group 2. RFC 5996

Group 2.1. Rekeying IKE SAs Using a CREATE_HLD_SA Exchange

[Test IKEv2.SGW.R.2.2.1.1.XXX: Sending INFORMATIONAL Exchange]

Purpose:

To verify an IKEv2 device can handle a particular type of address that the device does not support

References:

- [RFC 5996] - Section 3.15.4

Test Setup:

- Network Topology
  Connect the devices according to the Common Topology.
- Configuration
  In each part, configure the devices according to the Common Configuration.
- Pre-Sequence and Cleanup Sequence
  IKEv2 on the NUT is disabled after each part.

Procedure:

<table>
<thead>
<tr>
<th>Packet #1</th>
<th>See Common Packet #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration payload has 2 Configuration Attributes.</td>
<td></td>
</tr>
<tr>
<td>One is Attribute Type is INTERNAL_IP4_Address.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Packet #2</th>
<th>See Common Packet #5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Another is Attribute Type is INTERNAL_IP6_Address.</td>
<td></td>
</tr>
</tbody>
</table>

Part A: (ADVANCED)

1. TN1 transmits an IKE_SA_INIT request to NUT.
2. Observe the messages transmitted on Link A.
3. TN1 transmits an IKE_SA_INIT request to NUT. The request has 2 Configuration
Attributes. One is Attribute Type of INTERNAL_IP4_ADDRESS. Another is Attribute Type of INTERNAL_IP6_ADDRESS.

4. Observe the messages transmitted on Link A.

**Observable Results:**

**Part A**

**Step 2: Judgment #1**
The NUT transmits an IKE_SA_INIT request including "ENCR_3DES", "PRF_HMAC_SHA1", "AUTH_HMAC_SHA1_96" and "D-H Group 2" as proposed algorithms.

**Step 4: Judgment #2**
The NUT transmits an IKE_AUTH request including "ENCR_3DES", "AUTH_HMAC_SHA1_96" and "No Extended Sequence Numbers" as proposed algorithms. Configuration Payload is a Configuration Attribute. Its Attribute Type is INTERNAL_IP6_ADDRESS.

**Possible Problems:**

- If a device cannot discontinue support of IPv4 this test may be omitted.
All Rights Reserved. Copyright (C) 2008
Yokogawa Electric Corporation
Nippon Telegraph and Telephone Corporation (NTT)

No part of this documentation may be reproduced for any purpose without prior permission.