IKE Guidelines for Implementation and Priorities in Testing

version １.０.２

IPv6 Forum
IPv6 Promotion Council
Certification WG
IPsec SWG

****

This documentation is produced by IPsec SWG members of Certification WG in IPv6 Promotion Council.

The SWG members currently include Yokogawa Electric Corporation, NIPPON TELEGRAPH AND TELEPHONE CORPORATION (NTT).

Modification Record

Version 1.0.0	July 25, 2005	Initial version
Version 1.0.1	August 31, 2005	Editrial Fix.
Version 1.0.2	November 22, 2005	-Add “sending multiple proposal” as priority A2. -Modify “Kilobytes of SA Life Type” to “not supported” -Add a item to crarify commit-bit in RFC2408 section 3.1.

Table of Contents

1. Overview

This document gives guidelines for implementing the functions specified in the IETF RFC on the functions of IKE.

This document is provided

- as a guide to implementation that ensures interoperability between the End-Nodes, between the Security Gateways (SGWs), or between the Security Gateway (SGW) and End-Node,
- to give a classification of individual IKE functions according to their importance in terms of interoperability.

The IKE Test Profile consists of two volumes, [1] Guidelines for Implementation and Priorities in Testing (this document) and [2] Test Specifications.

The contents of this document include specifications of the interfaces between the nodes supporting IKE (i.e. SGW and End-Node), guidelines for the implementation of the nodes supporting IKE, and priorities for the testing of each node function according to the function's importance to interoperability.

This document is in complete accord with the IETF RFC specifications for IKE but includes some extra information for clarification and thus more strongly ensures interoperability.

Term Description

-End-Node

IPv6 node including a router that uses IKE to communicate of oneself

-Security Gateway

IPv6 node including a router or a firewall that intermediate system implementing IKE protocols.

2. Scope of the IKE Guidelines for Implementation and the test function it provides

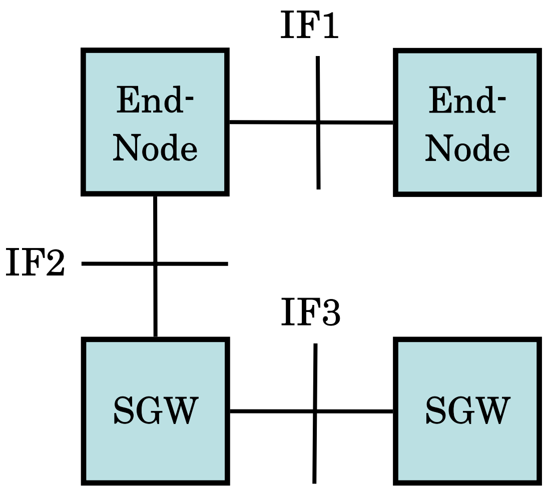
2.1 Reference Network Architecture

Figure 2-1 shows the network architecture covered by IKE Guidelines for Implementation.

- I/F1 is an interface that showed the protocol confirmation between End-Node and End-Node.

- I/F2 is an interface that showed the protocol confirmation between End-Node and Security Gateway.

- I/F3 is an interface that showed the protocol confirmation between Security Gateway and Security Gateway.



SGW: Security Gateway

Figure 2-1 Reference Network Architecture

This document only covers IKE specifications. Testing of generic IPv6 functions is beyond the scope of this test; however; some of the generic IPv6 functions are necessary to IKE functions and are thus supported in this test.

2.2 Related standards

This document covers the functions specified in the following IETF RFCs.

(1) RFC2407 (http://www.ietf.org/rfc/rfc2407.txt)

(2) RFC2408 (http://www.ietf.org/rfc/rfc2408.txt)

(3) RFC2409 (http://www.ietf.org/rfc/rfc2409.txt)

(4) RFC2401 (http://www.ietf.org/rfc/rfc2401.txt)

(5) RFC4109 (http://www.ietf.org/rfc/rfc4109.txt)

2.3 Classification of IKE functions needed for interoperability and provided as test function

This section describes ways to classify the IKE functions needed for interoperability and provided as test functions in the IKE Conformance Test.

2.3.1 Viewpoints of the classification

The classification of IKE functions is from the following viewpoints.

(A) IETF specification

(B) Functional Rank

(C) Test Priority

(A) IETF specification

IETF specification refers to the classification of each of the IKE functions from the viewpoint of importance for implementation as indicated by usage of the keywords below in the RFCs.

The keywords “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” are defined in RFC 2119

(B) Functional Rank

Functional Rank refers to classification of functions according to their importance to interoperability.

This classification is also based on descriptions in the IETF RFC; that is, functions with descriptions “MUST”, “SHOULD”, “MUST NOT”, and “SHOULD NOT” are basically classified as Rank-A, and functions with “MAY” are classified as Rank B, according to their importance to interoperability.

Table 2-1 shows the definition of Functional Rank.

Table 2-1 Definitions of Functional Rank

	Definitions of Functional Rank
Rank-A	These functions are essential to interoperability and should basically be implemented
Rank-B	Implementation of these functions is optional

Moreover, about the IKE function described on RFC except Keyword of above

MUST, SHOULD, and MAY, it is regarded as "do" (the role of a certain function is played), and distributed to Rank A or Rank B in consideration of the importance to interoperability from the above-mentioned table 2-1.

Furthermore, although not clearly written on RFC, the IKE function bundled as a supplementary matter in consideration of implementation is positioned as "add", and Functional Rank is assigned from the above-mentioned table 2-1.

(C) Test Priority

Test Priority is the classification from the viewpoint of the importance of testing.

Testing of the functions classified as Priority 1 is included in the minimum test package, for the testing of functions which are essential to interoperability.

Testing of the functions classified as Priority 2 are optional; this depends on the application to be used. The testing of Priority 2 (Optional Test) items is selectively incorporated in the test package according to the functions to be supported by the End-Node / SGW.

The functions assigned Rank A above are basically classified as Priority 1, however; some of the Rank A functions, i.e. those which are not always implemented, should be classified as Priority 2. All functions with Rank B are “Not Supported” by version 0.1.

Moreover, using the view of Functional Rank and Test Priority, the object which collected Rank A and Priority 1 is set to "A1."

The object which collected Rank A and Priority 2 similarly is set to "A2."

Since Rank B is “Not Supported”, it is classified as "B."

As a result, Functional Rank A was classified into Priority 1 and Priority 2.

Furthermore, about some functions, two Priorities may exist according to the kind of node (e.g. End-Node or SGW).

Refer to the table of Chapter 5 for the details of each classified function.

The reason is also described when two Priorities exist in the table.

Table 2-2 gives the definitions of Test Priority.

Table 2-2 Definitions of Test Priority

	Definitions of Test Priority
Priority 1 (Required Test)	Testing of the functions classified as Priority1 is included in the minimum test package, for the testing of functions that are essential to interoperability.
Priority 2 (Optional Test)	Testing of the functions classified as Priority2 may not be needed; this depends on the application to be used. The testing of Priority2 (Optional Test) items is selectively incorporated in the test package according to the functions to be supported by the End-Node / SGW.

2.3.2 Relationships among the classifications of functions and test items

Table 2-3 shows relationships among the classifications of functions and test items and coverage by this document. In consideration of the actual implementation and the direction of the marcket, however, there are some exceptions to table 2-3(e.g. a certain function of Priority 2 is "Not Supported").

Table 2-3 Classifications of and coverage by version 0.1 of the IKE Conformance Test

(A) IETF	(B) Functional Rank	(C) Test Priority
MUST MUST NOT	Rank-A	Priority 1 (Required Test)
SHOULD SHOULD NOT		Priority 2 (Optional Test)
MAY	Rank-B	Not Supported
do	Rank-A / Rank-B	Priority 1 / Priority 2
add	Rank-A / Rank-B	Priority 1 / Priority 2

🟩 supported by version 0.1 except some functions of Priority 2

🟨 not supported by version 0.1

As reference, the classification of Priority 1, Priority 2 and Not Supported is described for every node about a typical IKE function to the following table 2-4 to 2-6.

*The support of each function means the following.

- The node exchange parameters by IKE exchange.
- The node communicate by using exchanged parameters.

Table 2-4 IKE functions and its classifications for End-Node

Function		End-Node		
		Priority 1	Priority 2	Not Supported
IKE Phase1	Message Exchange Type	Main mode	Aggressive mode	New Group mode
	Initiator or Responder	Initiator, Responder	-	-
	Sending multiple proposal	-	Supported	-
ISAKMP SA	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC	
	Hash Algorithm	SHA1	MD5	Tiger
	Authentication Method	Pre-shared Key	Digital Signature （RSA）	Public key encryption, revised mode of public key encryption
	Diffie Hellman Group	Gourp2	Group1,5,14	Croup 3,4
	SA Life Type	Seconds	-	Kilobytes
IKE Phase2	Message Exchange Type	Quick mode	-	New Group mode
	Initiator or Responder	Initiator, Responder	-	-
	PFS	-	Supported	-
	Commit bit	-	Supported	-
	Re-key	Supported	-	
	Sending multiple proposal	-	Supported	-

IPsec SA	Encapsulation mode	Transport	Tunnel	-
	Security Protocol	ESP Auth	ESP	AH
	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC, NULL	
	Authentication Algorithm	HMAC-SHA1	HMAC-MD5, AES-XCBC	-
	SA Life Type	Seconds	-	Kilobytes
IPsec Communication	Encapsulation mode	Transport	Tunnel	-
	Security Protocol	ESP Auth	ESP	AH
	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC, NULL	
	Authentication Algorithm	HMAC-SHA1	HMAC-MD5, AES-XCBC	-
	Anti-replay	Sender node	Receiver node	-

Table 2-5 IKE functions and its classifications for SGW

Function		SGW		
		Priority 1	Priority 2	Not Supported
IKE Phase1	Message Exchange Type	Main mode	Aggressive mode	New Group mode
	Initiator or Responder	Initiator, Responder	-	-
	Sending multiple proposal	-	Supported	-
ISAKMP SA	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC	

	Hash Algorithm	SHA1	MD5	Tiger
	Authentication Method	Pre-shared Key	Digital Signature （RSA）	Public key encryption, revised mode of public key encryption
	Diffie Hellman Group	Gourp2	Group1,5,14	Croup 3,4
	SA Life Type	Seconds	-	Kilobytes
IKE Phase2	Message Exchange Type	Quick mode	-	New Group mode
	Initiator or Responder	Initiator, Responder	-	-
	PFS	-	Supported	-
	Commit bit	-	Supported	-
	Re-key	Supported	-	
	Sending multiple proposal	-	Supported	-
IPsec SA	Encapsulation mode	Tunnel	-	-
	Security Protocol	ESP Auth	ESP	AH
	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC, NULL	
	Authentication Algorithm	HMAC-SHA1	HMAC-MD5, AES-XCBC	-
	SA Life Type	Seconds	-	Kilobytes
IPsec Communication	Encapsulation mode	Tunnel	-	-
	Security Protocol	ESP Auth	ESP	AH
	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC, NULL	

	Authentication Algorithm	HMAC-SHA1	HMAC-MD5, AES-XCBC	-
	Anti-replay	Sender node	Receiver node	-

Table 2-6 IKE functions and its classifications for Mobile IPv6

Function		Mobile IPv6		
		Priority 1	Priority 2	Not Supported
IKE Phase1	Message Exchange Type	Aggressive mode	Main mode (Digital Signature)	New Group mode
	Initiator or Responder	MN:Initiator, HA:Responder	-	HA:Initiator, MN:Responder
	Sending multiple proposal	-	Supported	-
ISAKMP SA	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC	
	Hash Algorithm	SHA1	MD5	Tiger
	Authentication Method	Pre-shared Key	Digital Signature （RSA）	Public key encryption, revised mode of public key encryption
	Diffie Hellman Group	Gourp2	Group1,5,14	Croup 3,4
	SA Life Type	Seconds	-	Kilobytes
IKE Phase2	PFS	-	Supported	-
	Commit bit	-	Supported	-
	Re-key	Supported	-	-
	Sending multiple proposal	-	Supported	-

IPsec SA	Encapsulation mode	Transport	Tunnel*	-
	Security Protocol	ESP Auth	ESP	AH
	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC, NULL	
	Authentication Algorithm	HMAC-SHA1	HMAC-MD5, AES-XCBC	-
	SA Life Type	Seconds	-	Kilobytes
IPsec Communication	Encapsulation mode	Transport	Tunnel	-
	Security Protocol	ESP Auth	ESP	AH
	Encryption Algorithm	3DES-CBC	DES-CBC, AES128-CBC, NULL	
	Authentication Algorithm	HMAC-SHA1	HMAC-MD5, AES-XCBC	-
	Anti-replay	Sender node	Receiver node	-

*Tunnel is classified into Priority 2 because HoTI/HoT is classified into Priority 2 in Mobile IPv6 Guidelines.

3. Sequences

This section describes the IKE sequences used in the IKE Guidelines for Implementation. Sequences of test packets are sent to the target and expects to receive corresponding acknowledgement packets from the target. Details of the test sequences utilized in each test are given in the Test Specification documents.

The reference IKE sequences are shown from Figure 3-1 to Figure 3-3.

The actual sequences in which IKE runs are shown from Figure 3-4 to Figure 3-7.

The sequences in which rekey runs are shown from Figure 3-8 to Figure 3-11.

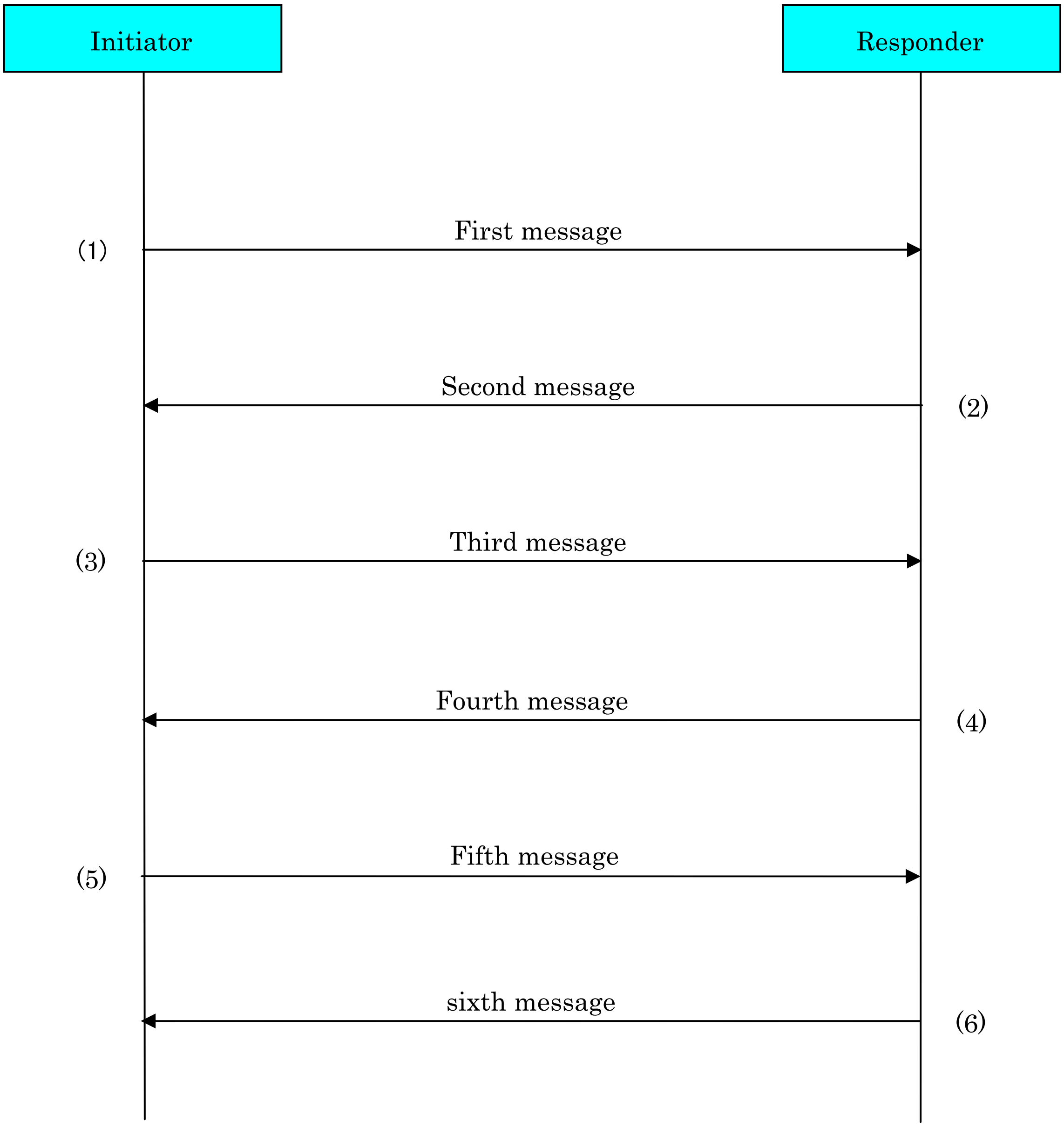


Figure3-1 IKE Phase1 Main Mode

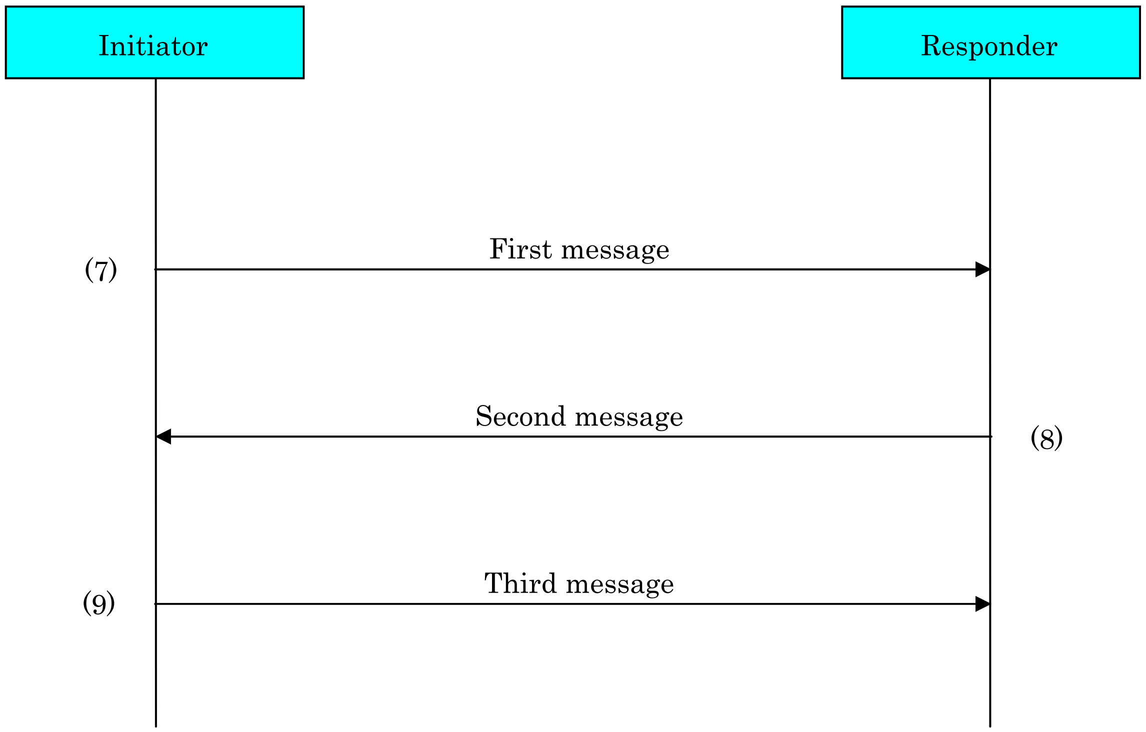


Figure3-2 IKE Phase1 Aggressive Mode

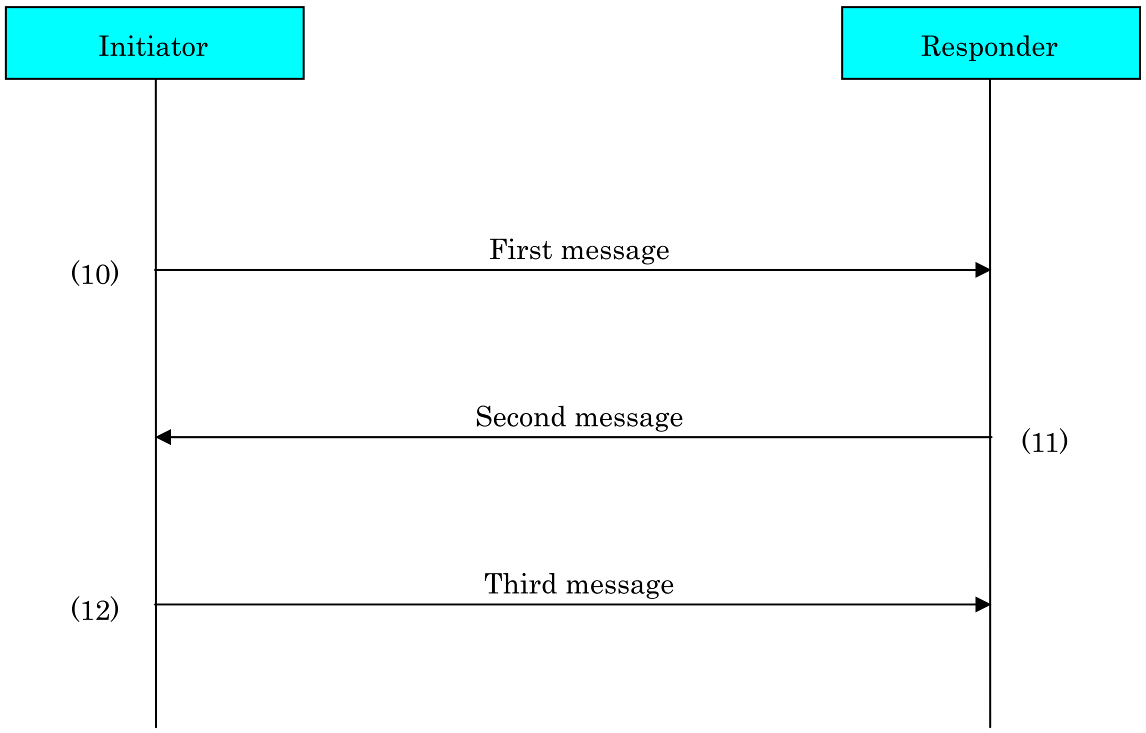


Figure3-3 IKE Phase 2 Quick Mode

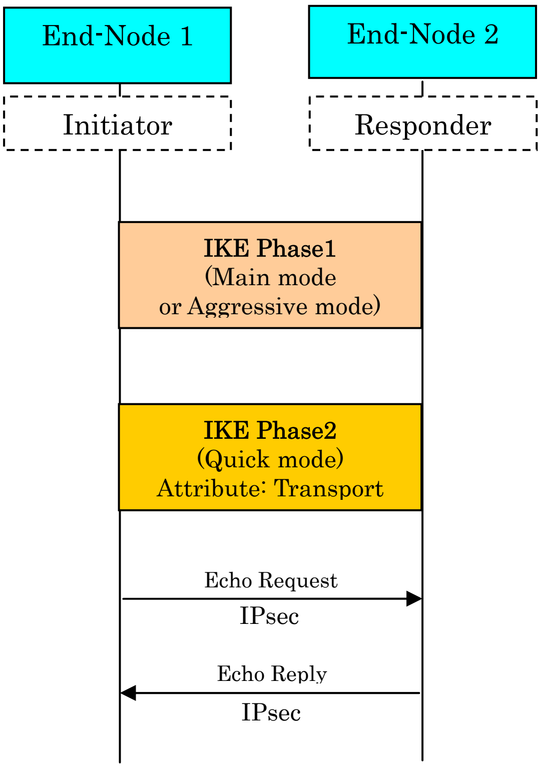


Figure3-4 End-Node to End-Node

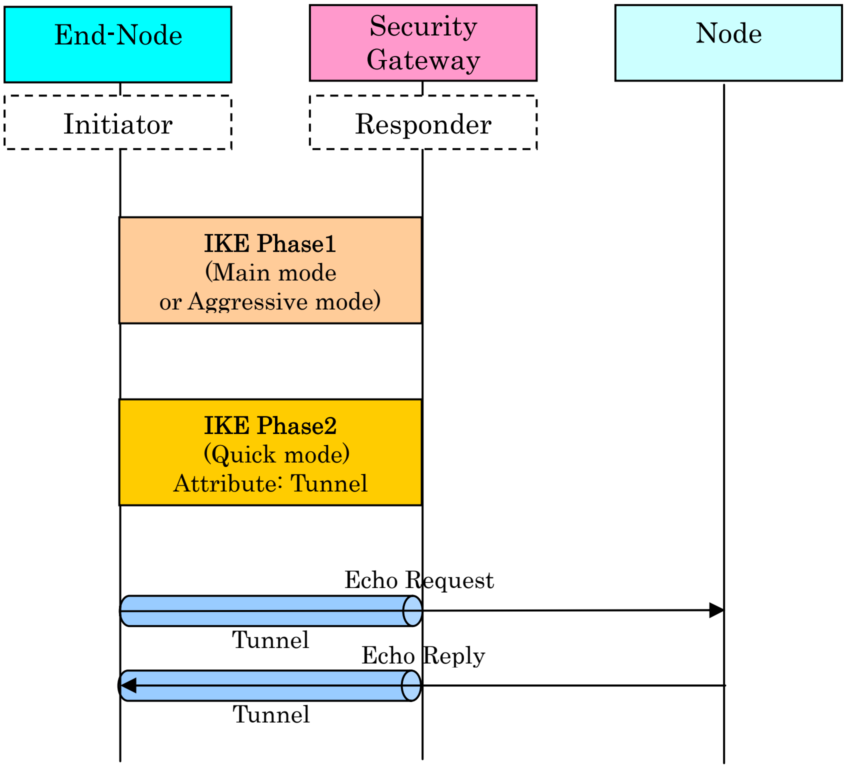


Figure3-5 End-Node to Security Gateway

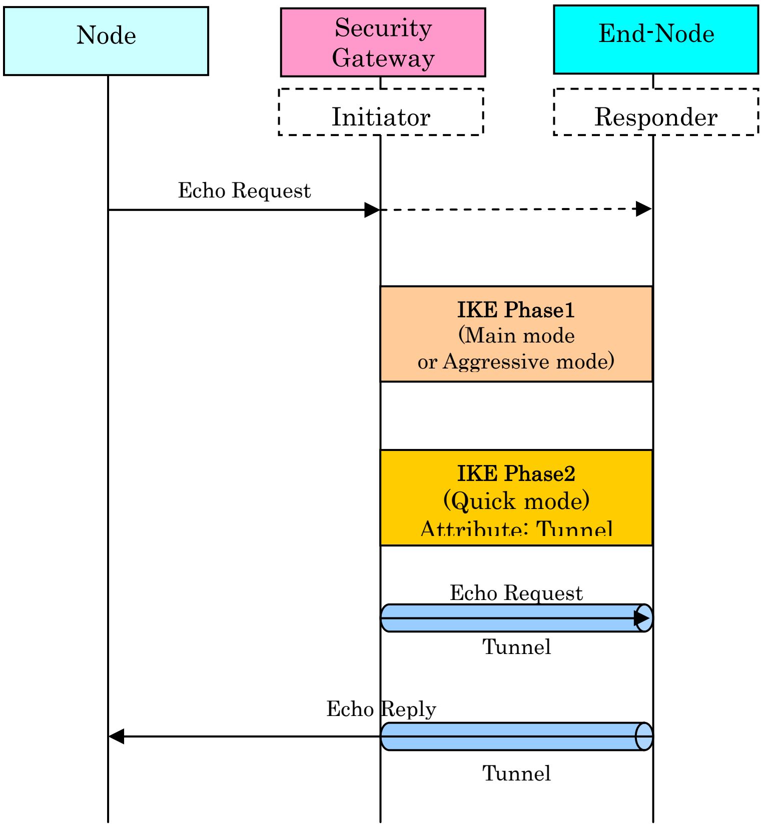


Figure3-6 Security Gateway to End-Node

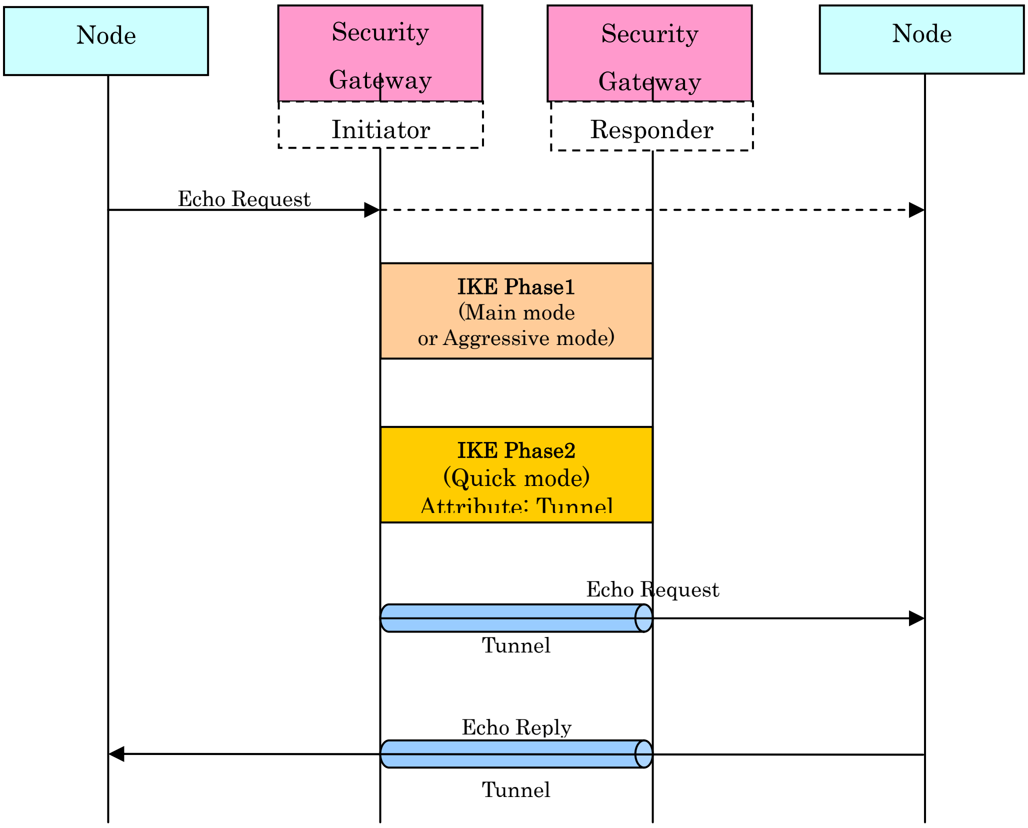


Figure3-7 Security Gateway to Security Gateway

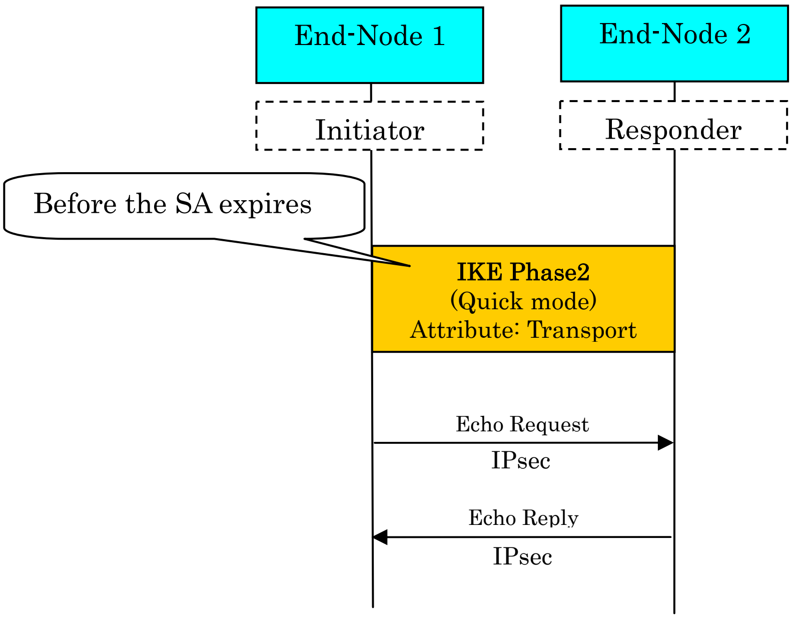


Figure3-8 Rekey by End-Node to End-Node

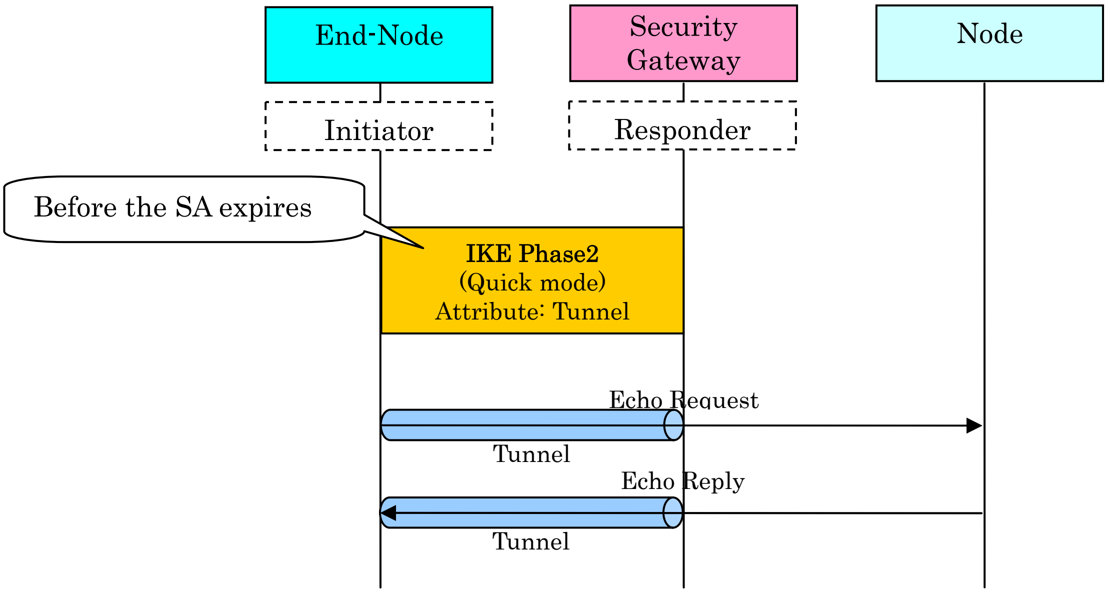


Figure3-9 Rekey by End-Node to Security Gateway

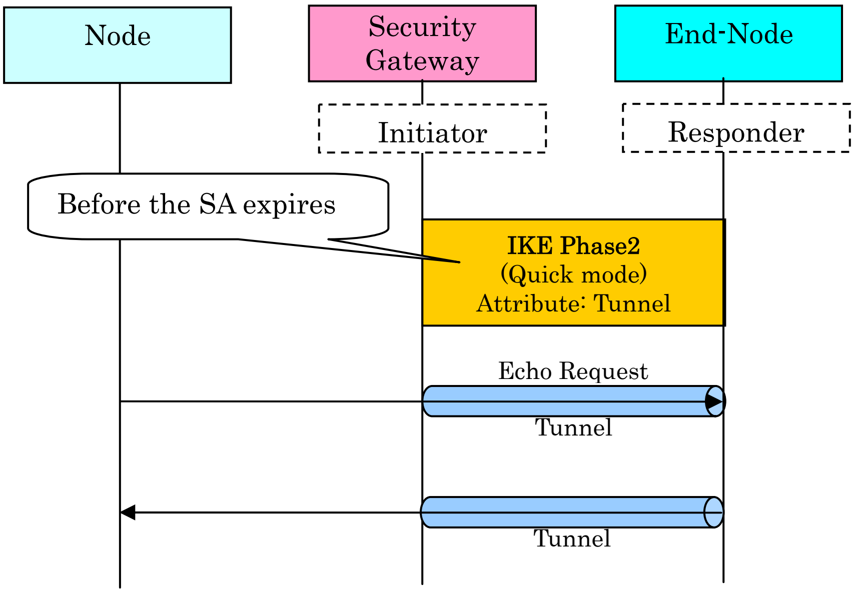


Figure3-10 Rekey by Security Gateway to End-Node

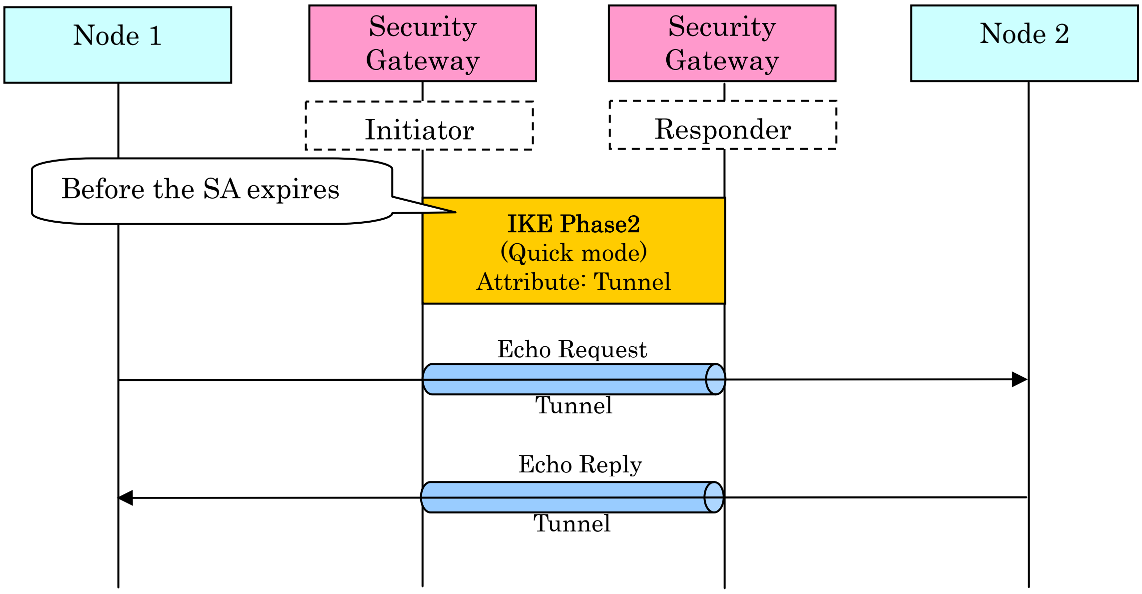


Figure3-11 Rekey by Security Gateway to Security Gateway

4. Packet formats

This section describes the references IKE packet formats which are utilized in the test sequences shown in section 3. IKE Conformance Test sends packets in these formats to the target and expects to receive the corresponding acknowledgement packets in these formats from the target. Details of the packet formats are given in the Test Specification documents.

A gray part means the encrypted packet in the following figures.

4.1. Phase1 Pre-shared key Main mode

(1)IKE Phase1 Pre-shared key Main Mode first message (Initiator -> Responder)

<table>
<tr><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Initiator 128bit)</td></tr>
<tr><td colspan="32">Destination Address(Responder 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie==random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=0</td></tr>
<tr><td colspan="8">Next Payload=1</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">●
Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=0</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Domain of Interpretation=1(IPsecDOI)</td></tr>
<tr><td colspan="32">Situation=1(SIT_IDENTITY_ONLY)</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">Proposal Number</td><td colspan="8">Protocol-ID=1(ISAKMP)</td><td colspan="8">SPI Size=0</td><td colspan="8">Number of Transform</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">Transform Number</td><td colspan="8">Transform-ID=1(KEY-IKE)</td><td colspan="16">Reserved2=0</td></tr>
<tr><td colspan="32">SA Attributes</td></tr>
</table>

(2)IKE Phase1 Pre-shared key Main Mode second message (Responder -> Initiator)

<table>
<tr><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>1</th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>2</th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>3</th><th></th></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Responder 128bit)</td></tr>
<tr><td colspan="32">Destination Address(Initiator 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie==random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=1</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=0</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Domain of Interpretation=1(IPsecDOI)</td></tr>
<tr><td colspan="32">Situation=1(SIT_IDENTITY_ONLY)</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">Proposal Number</td><td colspan="8">Protocol-ID=1(ISAKMP)</td><td colspan="8">SPI Size=0</td><td colspan="8">Number of Transform</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">Transform Number</td><td colspan="8">Transform-ID=1(KEY-IKE)</td><td colspan="16">Reserved2=0</td></tr>
<tr><td colspan="32">SA Attributes</td></tr>
</table>

(3)IKE Phase1 Pre-shared key Main Mode third message (Initiator -> Responder)

<table>
<tr><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Initiator 128bit)</td></tr>
<tr><td colspan="32">Destination Address(Responder 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie=random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=4</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=0</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=10</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Key Exchange Data</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Nonce Data</td></tr>
</table>

(4)IKE Phase1 Pre-shared key Main Mode firth message (Responder -> Initiator)

<table>
<tr><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Responder 128bit)</td></tr>
<tr><td colspan="32">Destination Address(Initiator 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie==random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=4</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">●</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=0</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=10</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Key Exchange Data</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Nonce Data</td></tr>
</table>

Reserved=0

(5)IKE Phase1 Pre-shared key Main Mode fifth message (Initiator -> Responder)

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Initiator 128bit)</td></tr>
<tr><td colspan="32">Destination Address(Responder 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie=random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=5</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">●
Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=1</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=8</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">ID Type=5</td><td colspan="8">Protocol ID=17</td><td colspan="16">Port=500</td></tr>
<tr><td colspan="32">Identification Data</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Hash Data(SHA1)=160bit</td></tr>
</table>

(6)IKE Phase1 Pre-shared key Main Mode sixth message (Initiator -> Responder)

<table>
<tr><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Responder 128bit)</td></tr>
<tr><td colspan="32">Destination Address(Initiator 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie==random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=5</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">●
Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=1</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=8</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">ID Type=5</td><td colspan="8">Protocol ID=17</td><td colspan="16">Port=500</td></tr>
<tr><td colspan="32">Identification Data</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Hash Data
(SHA1)=160bit</td></tr>
</table>

4.2. Phase1 Pre-shared key Aggressive mode

(7)IKE Phase1 Pre-shared key Aggressive Mode first message (Initiator-> Responder)

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Initiator128bit)</td></tr>
<tr><td colspan="32">Destination Address(Responder 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie==random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=0</td></tr>
<tr><td colspan="8">Next Payload=1</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=0</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=4</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Domain of Interpretation=1(IPsecDOI)</td></tr>
<tr><td colspan="32">Situation=1(SIT_IDENTITY_ONLY)</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">Proposal Number</td><td colspan="8">Protocol-ID=1(ISAKMP)</td><td colspan="8">SPI Size=0</td><td colspan="8">Number of Transform</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">Transform Number</td><td colspan="8">Transform-ID=1(KEY-IKE)</td><td colspan="16">Reserved2=0</td></tr>
<tr><td colspan="32">SA Attributes</td></tr>
</table>

Next Payload=10	Reserved=0	Payload Length
Key Exchange Data		
Next Payload=5	Reserved=0	Payload Length
Nonce Data		
Next Payload=0	Reserved=0	Payload Length
ID Type=5	Protocol ID=17	Port=500
Identification Data		

(8)IKE Phase1 Pre-shared key Aggressive Mode second message (Responder -> End-Node)

<table>
<tr><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Responder 128bit)</td></tr>
<tr><td colspan="32">Destination Address(Initiator128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie==random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=1</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=0</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=4</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Domain of Interpretation=1(IPsecDOI)</td></tr>
<tr><td colspan="32">Situation=1(SIT_IDENTITY_ONLY)</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">Proposal Number</td><td colspan="8">Protocol-ID=1(ISAKMP)</td><td colspan="8">SPI Size=0</td><td colspan="8">Number of Transform</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="8">Transform Number</td><td colspan="8">Transform-ID=1(KEY-IKE)</td><td colspan="16">Reserved2=0</td></tr>
<tr><td colspan="32">SA Attributes</td></tr>
</table>

Next Payload=10	Reserved=0	Payload Length
Key Exchange Data		
Next Payload=5	Reserved=0	Payload Length
Nonce Data		
Next Payload=8	Reserved=0	Payload Length
ID Type=5	Protocol ID=17	Port=500
Identification Data		
Next Payload=0	Reserved=0	Payload Length
Hash Data(SHA1)=160bit		

(9)IKE Phase1 Pre-shared key Aggressive Mode third message (Initiator-> Responder)

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Initiator128bit)</td></tr>
<tr><td colspan="32">Destination Address(Responder 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie=random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=8</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=2</td><td colspan="2">●</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=1</td></tr>
<tr><td colspan="32">Message ID=0</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=0</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Hash Data(SHA1)=160bit</td></tr>
</table>

Reserved=0

4.3. Phase2 Quick mode

(10) IKE Phase 2 Quick Mode first message (Initiator-> Responder)

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Initiator128bit)</td></tr>
<tr><td colspan="32">Destination Address(Responder 128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie=random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=8</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=32</td><td colspan="2">●
Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=1</td></tr>
<tr><td colspan="32">Message ID=random(ZZZ)</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=1</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Hash Data(SHA1)=160bit</td></tr>
<tr><td colspan="8">Next Payload=10</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Domain of Interpretation=1(IPsecDOI)</td></tr>
<tr><td colspan="32">Situation=1(SIT_IDENTITY_ONLY)</td></tr>
</table>

Next Payload=0	Reserved=0	Payload Length		
Proposal Number	Protocol-ID=3	SPI Size=4	Number of Transform	
SPI(32bit)				
Next Payload=0	Reserved=0	Payload Length		
Transform Number	Transform-ID=2(ESP-DES)	Reserved2=0		
SA Attributes				
Next Payload=5	Reserved=0	Payload Length		
Nonce Data				
Next Payload=5	Reserved=0	Payload Length		
ID Type=5	Protocol ID=0	Port=0		
Identification Data (Initiator)				
Next Payload=0	Reserved=0	Payload Length		Option
ID Type=5	Protocol ID=0	Port=0		
Identification Data (Responder)				

(11) IKE Phase 2 Quick Mode second message (Responder -> End-Node)

<table>
<tr><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td colspan="4">Ver=6</td><td colspan="8">Traffic Class</td><td colspan="20">Flow Label</td></tr>
<tr><td colspan="16">Payload Length</td><td colspan="8">Next Header=17</td><td colspan="8">Hop Limit</td></tr>
<tr><td colspan="32">Source Address(Responder 128bit)</td></tr>
<tr><td colspan="32">Destination Address(Initiator128bit)</td></tr>
<tr><td colspan="16">Source Port=500</td><td colspan="16">Destination Port=500</td></tr>
<tr><td colspan="16">Length</td><td colspan="16">Checksum</td></tr>
<tr><td colspan="32">Initiator Cookie=random(XXX)</td></tr>
<tr><td colspan="32">Responder Cookie=random(YYY)</td></tr>
<tr><td colspan="8">Next Payload=8</td><td colspan="4">MjVer=1</td><td colspan="4">MnVer=0</td><td colspan="8">Exchange Type=32</td><td colspan="2">●
Reserved=0</td><td colspan="2">A=0</td><td colspan="2">C=0</td><td colspan="2">E=1</td></tr>
<tr><td colspan="32">Message ID=random(ZZZ)</td></tr>
<tr><td colspan="32">Length</td></tr>
<tr><td colspan="8">Next Payload=1</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Hash Data(SHA1)=160bit</td></tr>
<tr><td colspan="8">Next Payload=10</td><td colspan="8">Reserved=0</td><td colspan="16">Payload Length</td></tr>
<tr><td colspan="32">Domain of Interpretation=1(IPsecDOI)</td></tr>
<tr><td colspan="32">Situation=1(SIT_IDENTITY_ONLY)</td></tr>
</table>

Next Payload=0	Reserved=0	Payload Length	
Proposal Number	Protocol-ID=3	SPI Size=4	Number of Transform
SPI(32bit)			
Next Payload=0	Reserved=0	Payload Length	
Transform Number	Transform-ID=2(ESP-DES)	Reserved2=0	
SA Attributes			
Next Payload=5	Reserved=0	Payload Length	
Nonce Data			
Next Payload=5	Reserved=0	Payload Length	
ID Type=5	Protocol ID=0	Port=0	
Identification Data (Initiator)			
Next Payload=0	Reserved=0	Payload Length	
ID Type=5	Protocol ID=0	Port=0	
Identification Data (Responder)			

Option

(12) IKE Phase 2 Quick Mode third message (Initiator-> Responder)

```
                    1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
|  Ver=6  |   Traffic Class   |              Flow Label             |
|          Payload Length         | Next Header=17 |    Hop Limit    |
|                                                                   |
|             Source Address(Initiator128bit)                       |
|                                                                   |
|                                                                   |
|          Destination Address(Responder 128bit)                    |
|                                                                   |
|        Source Port=500          |      Destination Port=500       |
|            Length               |            Checksum             |
|                                                                   |
|               Initiator Cookie=random(XXX)                        |
|                                                                   |
|               Responder Cookie=random(YYY)                        |
| Next Payload=8 | MjVer=1 | MnVer=0 | Exchange Type=2 | ● |A=0|C=0|E=1|
|                  Message ID=random(ZZZ)                           |
|                          Length                                   |
| Next Payload=0 |   Reserved=0    |         Payload Length         |
|                                                                   |
|                 Hash Data(SHA1)=160bit                            |
|                                                                   |
```

● Reserved=0

4.4. SA Attributes

(a)IKE Phase1

(a-1) Encryption Algorithm

											1										2										3	
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Format=1	Type=1																Value=3(3DES-CBC)															

(a-2) Hash Algorithm

											1										2										3	
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Format=1	Type=2																Value=2(HMAC-SHA)															

(a-3) Authentication Method

											1										2										3	
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Format=1	Type=3																Value=1(pre-shared key)															

(a-4) Group Description

											1										2										3	
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Format=1	Type=4																Value=2(1024-bit MODP group)															

(a-5) SA Life Type

											1										2										3	
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Format=1	Type=0xB																Value=1(seconds)															

(b)IKE Phase2

(b-1) SA Life Type

<table>
<tr><td></td><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td>Format=1</td><td colspan="16">Type=1</td><td colspan="16">Value=1(seconds)</td></tr>
</table>

(b-2) Group Description

<table>
<tr><td></td><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td>Format=1</td><td colspan="16">Type=3</td><td colspan="16">Value=2(1024-bit MODP group)</td></tr>
</table>

(b-3) Encapsulation Mode

- Transport mode

<table>
<tr><td></td><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td>Format=1</td><td colspan="16">Type=4</td><td colspan="16">Value=2(Transport)</td></tr>
</table>

- Tunnel mode

<table>
<tr><td></td><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td>Format=1</td><td colspan="16">Type=4</td><td colspan="16">Value=1(Tunnel)</td></tr>
</table>

(b-4) Authentication Algorithm

<table>
<tr><td></td><td colspan="10"></td><td>1</td><td colspan="9"></td><td>2</td><td colspan="9"></td><td>3</td><td></td></tr>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td>Format=1</td><td colspan="16">Type=5</td><td colspan="16">Value=2(HAMC-SHA)</td></tr>
</table>

5. Functional classification and test priority for individual IPv6 nodes

This chapter describes the operation for IKE and the functional classifications on the basis of the classifications given in chapter 2.3.

Notes

- “RFC section” gives the corresponding section number in the RFC referred to in chapter 2.2.
- “RFC section title” gives the section heading in the RFC referred to in chapter 2.2.
- In the column “Test Priority,” “A1” indicates Rank A and Priority 1, “A2” indicates Rank-A and Priority 2, and “B” indicates Rank-B.
- “Reason of TEST Priority” gives the reason for the function’s classification. Basically, a reason is given when Test Priority is “A2” or ”B”.

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	4.2.1	SIT_IDENTITY_ONLY	SIT_IDENTITY_ONLY	All IPSEC DOI implementations **MUST** support SIT_IDENTITY_ONLY by including an Identification Payload in at least one of the Phase I Oakley exchanges ([IKE], Section 5) and **MUST** abort any association setup that does not include an Identification Payload.	MUST	A1			A1		
2					MUST	A1			A1		
3				If an initiator supports neither SIT_SECRECY nor SIT_INTEGRITY, the situation consists only of the 4 octet situation bitmap and does not include the Labeled Domain Identifier field (Figure 1, Section 4.6.1) or any subsequent label information. Conversely, if the initiator supports either SIT_SECRECY or SIT_INTEGRITY, the Labeled Domain Identifier **MUST** be included in the situation payload.	MUST	A1			A1		
4	4.2.2	SIT_SECRECY	SIT_SECRECY	If an initiator does not support SIT_SECRECY, SIT_SECRECY **MUST NOT** be set in the Situation bitmap and no secrecy level or category bitmaps shall be included.	MUST NOT	B		dependent on a support situation	B		dependent on a support situation
5				If a responder does not support SIT_SECRECY, a SITUATION-NOT-SUPPORTED Notification Payload **SHOULD** be returned and the security association setup **MUST** be aborted.	SHOULD	B		Notification Payload	B		Notification Payload
6					MUST	B		dependent on a support situation	B		dependent on a support situation
7	4.2.3	SIT_INTEGRITY	SIT_INTEGRITY	If an initiator does not support SIT_INTEGRITY, SIT_INTEGRITY **MUST NOT** be set in the Situation bitmap and no integrity level or category bitmaps shall be included.	MUST NOT	B		dependent on a support situation	B		dependent on a support situation
8				If a responder does not support SIT_INTEGRITY, a SITUATION-NOT-SUPPORTED Notification Payload **SHOULD** be returned and the security association setup **MUST** be aborted.	SHOULD	B		Notification Payload	B		Notification Payload
9					MUST	B		dependent on a support situation	B		dependent on a support situation

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	4.4.1.1	PROTO_ISAKMP	PROTO_ISAKMP	All implementations within the IPSEC DOI **MUST** support PROTO_ISAKMP.	MUST	A1			A1		
2	4.4.1.2	PROTO_IPSEC_AH	PROTO_IPSEC_AH	For export control considerations, confidentiality **MUST NOT** be provided by any PROTO_IPSEC_AH transform.	MUST NOT	B		AH	B		AH
3	4.4.2.1	KEY_IKE	KEY_IKE	All implementations within the IPSEC DOI **MUST** support KEY_IKE.	MUST	A1			A1		
4	4.4.3	IPSEC AH Transform Identifiers	AH	Note: the Authentication Algorithm attribute **MUST** be specified to identify the appropriate AH protection suite.	MUST	B		AH	B		AH
5				Note: all mandatory-to-implement algorithms are listed as "**MUST**" implement (e.g. AH_MD5) in the following sections. All other algorithms are optional and **MAY** be implemented in any particular implementation.	MUST	-		sentence of description	-		sentence of description
6					MAY	-		sentence of description	-		sentence of description
7	4.4.3.1	AH_MD5	AH_MD5	All implementations within the IPSEC DOI **MUST** support AH_MD5 along with the Auth(HMAC-MD5) attribute.	MUST	B		AH	B		AH
8	4.4.3.2	AH_SHA	AH_SHA	All implementations within the IPSEC DOI **MUST** support AH_SHA along with the Auth(HMAC-SHA) attribute.	MUST	B		AH	B		AH
9	4.4.3.3	AH_DES	AH_DES	The IPSEC DOI defines AH_DES along with the Auth(DES-MAC) attribute to be a DES-MAC transform. Implementations are not required to support this mode.	(do)	B		AH	B		AH
10	4.4.4	IPSEC ESP Transform Identifiers	ESP	Note: when authentication, integrity protection, and replay detection are required, the Authentication Algorithm attribute **MUST** be specified to identify the appropriate ESP protection suite.	MUST	A1			A1		
11				Note: all mandatory-to-implement algorithms are listed as "**MUST**" implement (e.g. ESP_DES) in the following sections. All other algorithms are optional and **MAY** be implemented in any particular implementation.	MUST	-		sentence of description	-		sentence of description
12					MAY	-		sentence of description	-		sentence of description
13	4.4.4.2	ESP_DES	ESP_DES	All implementations within the IPSEC DOI **MUST** support ESP_DES along with the Auth(HMAC-MD5) attribute.	MUST	A2		ESP-DES	A2		ESP-DES
14	4.4.4.3	ESP_3DES	ESP_3DES	All implementations within the IPSEC DOI are strongly encouraged to support ESP_3DES along with the Auth(HMAC-MD5) attribute.	(do)	A1			A1		
15	4.4.4.11	ESP_NULL	ESP_NULL	All implementations within the IPSEC DOI **MUST** support ESP_NULL. The ESP NULL transform is defined in [ESPNULL].	MUST	A2		ESP-NULL	A2		ESP-NULL
16	4.4.5.1	IPCOMP_OUI	IPCOMP_OUI	The IPCOMP_OUI type specifies a proprietary compression transform. The IPCOMP_OUI type must be accompanied by an attribute which further identifies the specific vendor algorithm.	(do)	B		IPCOMP	B		IPCOMP

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	4.5	IPSEC Security Association Attributes	SA Attributes	Attributes described as basic **MUST NOT** be encoded as variable. Variable length attributes **MAY** be encoded as basic attributes if their value can fit into two octets.	MUST NOT	A1			A1		
2					MAY	A1/B		A: Receiver C: Sender	A1/B		A: Receiver C: Sender
3			SA Duration	If unspecified, the default value shall be assumed to be 28800 seconds (8 hours).	(do)	A1			A1		
4				An SA Life Duration attribute **MUST** always follow an SA Life Type which describes the units of duration.	MUST	A1			A1		
5			Authentication Algorithm	When negotiating ESP without authentication, the Auth Algorithm attribute **MUST NOT** be included in the proposal.	MUST NOT	B		ESP without authentication	B		ESP without authentication
6				When negotiating ESP without confidentiality, the Auth Algorithm attribute **MUST** be included in the proposal and the ESP transform ID must be ESP_NULL.	MUST	B		ESP-NULL	B		ESP-NULL
7			Key Length	There is no default value for Key Length, as it must be specified for transforms using ciphers with variable key lengths. For fixed length ciphers, the Key Length attribute **MUST NOT** be sent.	MUST NOT	A1			A1		
8	4.5.1	Required Attribute Support	attributes	To ensure basic interoperability, all implementations **MUST** be prepared to negotiate all of the following attributes. SA Life Type SA Duration Auth Algorithm	MUST	A1			A1		
9	4.5.2	Attribute Parsing Requirement (Lifetime)		To allow for flexible semantics, the IPSEC DOI requires that a conforming ISAKMP implementation **MUST** correctly parse an attribute list that contains multiple instances of the same attribute class, so long as the different attribute entries do not conflict with one another.Currently, the only attributes which requires this treatment are Life Type and Duration.	MUST	A1			A1		
10				If conflicting attributes are detected, an ATTRIBUTES-NOT-SUPPORTED Notification Payload **SHOULD** be returned and the security association setup **MUST** be aborted.	SHOULD	B		Informational Exchange	B		Informational Exchange
11					MUST	A1			A1		
12	4.5.3	Attribute Negotiation		If an implementation receives a defined IPSEC DOI attribute (or attribute value) which it does not support, an ATTRIBUTES-NOT-SUPPORT **SHOULD** be sent and the security association setup **MUST** be aborted, unless the attribute value is in the reserved range.	SHOULD	B		Informational Exchange	B		Informational Exchange
13					MUST	A1			A1		
14				If an implementation receives an attribute value in the reserved range, an implementation **MAY** chose to continue based on local policy.	MAY	B		local policy	B		local policy

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
15	4.5.4	Lifetime Notification		When an initiator offers an SA lifetime greater than what the responder desires based on their local policy, the responder has three choices: 1) fail the negotiation entirely; 2) complete the negotiation but use a shorter lifetime than what was offered; 3) complete the negotiation and send an advisory notification to the initiator indicating the responder's true lifetime. The choice of what the responder actually does is implementation specific and/or based on local policy.	(do)	B		local policy	B		local policy
16				To ensure interoperability in the latter case, the IPSEC DOI requires the following only when the responder wishes to notify the initiator: if the initiator offers an SA lifetime longer than the responder is willing to accept, the responder **SHOULD** include an ISAKMP Notification Payload in the exchange that includes the responder's IPSEC SA payload. Section 4.6.3.1 defines the payload layout for the RESPONDER-LIFETIME Notification Message type which **MUST** be used for this purpose.	SHOULD	B		Notification Payload	B		Notification Payload
17					MUST	B		RESPONDER-LIFETIME	B		RESPONDER-LIFETIME

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	4.6.1	Security Association Payload	Secrecy Level	The secrecy level **MUST** be padded with zero (0) to align on the next 32-bit boundary.	MUST	B		not used in SIT-IDENTITY-ONLY	B		not used in SIT-IDENTITY-ONLY
2			Secrecy Category Bitmap	The bitmap **MUST** be padded with zero (0) to align on the next 32-bit boundary.	MUST	B		not used in SIT-IDENTITY-ONLY	B		not used in SIT-IDENTITY-ONLY
3			Integrity Level	The integrity level **MUST** be padded with zero (0) to align on the next 32-bit boundary.	MUST	B		not used in SIT-IDENTITY-ONLY	B		not used in SIT-IDENTITY-ONLY
4			Integrity Category Bitmap	The bitmap **MUST** be padded with zero (0) to align on the next 32-bit boundary.	MUST	B		not used in SIT-IDENTITY-ONLY	B		not used in SIT-IDENTITY-ONLY
5	4.6.2	Identification Payload Content	The identity of the initiator	The identity of the initiator **SHOULD** be used by the responder to determine the correct host system security policy requirement for the association.	SHOULD	A1			A1		
6			ID port and protocol fields during Phase I negotiations	During Phase I negotiations, the ID port and protocol fields **MUST** be set to zero or to UDP port 500. If an implementation receives any other values, this **MUST** be treated as an error and the security association setup **MUST** be aborted. This event **SHOULD** be auditable.	MUST	A1			A1		
7					MUST	A1			A1		
8					MUST	A1			A1		
9					SHOULD	B		logging	B		logging
10			Protocol ID	A value of zero means that the Protocol ID field should be ignored.	(do)	A1			A1		
11			Port	Value specifying an associated port. A value of zero means that the Port field should be ignored.	(do)	A1			A1		
12	4.6.2.1	Identification Type Values	length	For types where the ID entity is variable length, the size of the ID entity is computed from size in the ID payload header.	(do)	A1			A1		
13			certificates	When an IKE exchange is authenticated using certificates (of any format), any ID's used for input to local policy decisions **SHOULD** be contained in the certificate used in the authentication of the exchange.	SHOULD	A2		certificates	A2		certificates
14	4.6.3	IPSEC Notify Message Types	Notification Status Message Types	Notification Status Messages **MUST** be sent under the protection of an ISAKMP SA: either as a payload in the last Main Mode exchange; in a separate Informational Exchange after Main Mode or Aggressive Mode processing is complete; or as a payload in any Quick Mode exchange.These messages **MUST NOT** be sent in Aggressive Mode exchange, since Aggressive Mode does not provide the necessary protection to bind the Notify Status Message to the exchange.	MUST	B		Notify Message Types	B		Notify Message Types
15					MUST NOT	B		Notify Message Types	B		Notify Message Types
16				To ensure receipt of any particular message, the sender **SHOULD** include a Notification Payload in a defined Main Mode or Quick Mode exchange which is protected by a retransmission timer.	SHOULD	B		Notify Message Types	B		Notify Message Types

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
17	4.6.3.1	RESPONDER-LIFETIME	the format of Notification Payload	o Payload Length - set to length of payload + size of data (var) o DOI - set to IPSEC DOI (1) o Protocol ID - set to selected Protocol ID from chosen SA o SPI Size - set to either sixteen (16) (two eight-octet ISAKMP cookies) or four (4) (one IPSEC SPI) o Notify Message Type - set to RESPONDER-LIFETIME (Section 4.6.3) o SPI - set to the two ISAKMP cookies or to the sender's inbound IPSEC SPI o Notification Data - contains an ISAKMP attribute list with the	MUST	B		Notify Message Types	B		Notify Message Types
18	4.6.3.2	REPLAY-STATUS	the format of Notification Payload	o Payload Length - set to length of payload + size of data (4) o DOI - set to IPSEC DOI (1) o Protocol ID - set to selected Protocol ID from chosen SA o SPI Size - set to either sixteen (16) (two eight-octet ISAKMP cookies) or four (4) (one IPSEC SPI) o Notify Message Type - set to REPLAY-STATUS o SPI - set to the two ISAKMP cookies or to the sender's inbound IPSEC SPI o Notification Data - a 4 octet value: 0 = replay detection disabled 1 = replay detection enabled	MUST	B		Notify Message Types	B		Notify Message Types
19	4.6.3.3	INITIAL-CONTACT	INITIAL-CONTACT status message	The receiver of this Notification Message might then elect to delete any existing SA's it has for the sending system under the assumption that the sending system has rebooted and no longer has access to the original SA's and their associated keying material. When used, the content of the Notification Data field **SHOULD** be null (i.e. the Payload Length should be set to the fixed length of Notification	SHOULD	B		Notify Message Types	B		Notify Message Types
20			the format of Notification Payload	o Payload Length - set to length of payload + size of data (0) o DOI - set to IPSEC DOI (1) o Protocol ID - set to selected Protocol ID from chosen SA o SPI Size - set to sixteen (16) (two eight-octet ISAKMP cookies) o Notify Message Type - set to INITIAL-CONTACT o SPI - set to the two ISAKMP cookies o Notification Data - <not included>	MUST	B		Notify Message Types	B		Notify Message Types

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	3.1	ISAKMP Header Format	o Major Version (4 bits)	Implementations based on this version of the ISAKMP Internet-Draft **MUST** set the Major Version to 1.Implementations based on previous versions of ISAKMP Internet-Drafts **MUST** set the Major Version to 0. Implementations **SHOULD** never accept packets with a major version number larger than its own.	MUST	A1			A1		
2					MUST	B		previous versions	B		previous versions
3					SHOULD	A2		new versions	A2		new versions
4			o Minor Version (4 bits)	Implementations based on this version of the ISAKMP Internet-Draft **MUST** set the Minor Version to 0.Implementations based on previous versions of ISAKMP Internet-Drafts **MUST** set the Minor Version to 1.Implementations **SHOULD** never accept packets with a minor version number larger than its own, given the major version numbers are identical.	MUST	A1			A1		
5					MUST	B		previous versions	B		previous versions
6					SHOULD	A2		new versions	A2		new versions
7			o Flags (1 octet)	The flags listed below are specified in the Flags field beginning with the least significant bit, i.e the Encryption bit is bit 0 of the Flags field, the Commit bit is bit 1 of the Flags field, and the Authentication Only bit is bit 2 of the Flags field.The remaining bits of the Flags field **MUST** be set to 0 prior to transmission.	MUST	A1			A1		
8			-- E(ncryption Bit) (1 bit)	If set (1), all payloads following the header are encrypted using the encryption algorithm identified in the ISAKMP SA. The ISAKMP SA Identifier is the combination of the initiator and responder cookie. It is RECOMMENDED that encryption of communications be done as soon as possible between the peers.For all ISAKMP exchanges described in section 4.1, the encryption **SHOULD** begin after both parties have exchanged Key Exchange payloads.If the E(ncryption Bit) is not set (0), the payloads are not encrypted.	SHOULD	A1			A1		
9			-- C(ommit Bit) (1 bit)	The Commit Bit can be set (at anytime) by either party participating in the SA establishment, and can be used during both phases of an ISAKMP SA establishment. However, the value **MUST** be reset after the Phase 1 negotiation.	MUST	A1			A1		
10				If set(1), the entity which did not set the Commit Bit **MUST** wait for an Informational Exchange containing a Notify payload (with the CONNECTED Notify Message) from the entity which set the Commit Bit. In this instance, the Message ID field of the Informational Exchange **MUST** contain the Message ID of the original ISAKMP Phase 2 SA negotiation.	MUST	A2		Commit Bits	A2		Commit Bit
11					MUST	A2		Commit Bits	A2		Commit Bit
12				It is always possible that the final message of an exchange can be lost. In this case, the entity expecting to receive the final message of an exchange would receive the Phase 2 SA negotiation message following a Phase 1 exchange or encrypted traffic following a Phase 2 exchange. Handling of this situation is not standardized, but we propose the following possibilities. If the entity awaiting the Informational Exchange can verify the received message (i.e. Phase 2 SA negotiation message or encrypted traffic), then they **MAY** consider the SA was established and continue processing.	MAY	B		awaiting the Informational Exchange	B		awaiting the Informational Exchange
13				Informational exchange with the CONNECTED Notify is sent as part of the Quick Mode exchange and not as a seperate Informational exchange. And initialization vector(IV) of informational exchange with the CONNECTED Notify is created by the last encryption block of the third Quick Mode message.	(add)	A2		Commit Bits	A2		Commit Bit

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
14			-- A(uthentication Only Bit) (1 bit)	This bit is intended for use with the Informational Exchange with a Notify payload and will allow the transmission of information with integrity checking, but no encryption (e.g. "emergency mode").Section 4.8 states that a Phase 2 Informational Exchange **MUST** be sent under the protection of an ISAKMP SA.	MUST	B		Authentication Only Bit	B		Authentication Only Bit
15			o Message ID (4 octets)	Unique Message Identifier used to identify protocol state during Phase 2 negotiations. This value is randomly generated by the initiator of the Phase 2 negotiation.	(do)	A1			A1		
16				During Phase 1 negotiations, the value **MUST** be set to 0.	MUST	A1			A1		
17	3.4	Security Association Payload	o Next Payload (1 octet)	Identifier for the payload type of the next payload in the message. If the current payload is the last in the message, then this field will be 0.This field **MUST** NOT contain the values for the Proposal or Transform payloads as they are considered part of the security association negotiation.	MUST	A1			A1		
18			o Domain of Interpretation (4 octets)	This field **MUST** be present within the Security Association payload.	MUST	A1			A1		
19			o Situation (variable length)	This field **MUST** be present within the Security Association payload.	MUST	A1			A1		
20	3.5	Proposal Payload	o Next Payload (1 octet)	This field **MUST** only contain the value "2" or "0".If there are additional Proposal payloads in the message, then this field will be 2. If the current Proposal payload is the last within the security association proposal, then this field will be 0.	MUST	A1			A1		
21			o SPI Size (1 octet)	In the case of ISAKMP, the Initiator and Responder cookie pair from the ISAKMP Header is the ISAKMP SPI, therefore, the SPI Size is irrelevant and **MAY** be from zero (0) to sixteen (16). If the SPI Size is non-zero, the content of the SPI field MUST be ignored.	MAY	A1			A1		
22					MUST	A1			A1		
23	3.6	Transform Payload	o Next Payload (1 octet)	This field **MUST** only contain the value "3" or "0".If there are additional Transform payloads in the proposal, then this field will be 3. If the current Transform payload is the last within the proposal, then this field will be 0.	MUST	A1			A1		
24			o SA Attributes (variable length)	The SA Attributes **SHOULD** be represented using the Data Attributes format described in section 3.3.If the SA Attributes are not aligned on 4-byte boundaries,then subsequent payloads will not be aligned and any padding will be added at the end of the message to make the message 4-octet aligned.	SHOULD	A1			A1		
25	3.8	Identification Payload	o DOI Specific ID Data (3 octets)	Contains DOI specific Identification data. If unused, then this field **MUST** be set to 0.	MUST	A1			A1		
26	3.9	Certificate Payload	certificate payloads	Certificate payloads **SHOULD** be included in an exchange whenever an appropriate directory service (e.g. Secure DNS [DNSSEC]) is not available to distribute certificates.	SHOULD	A2		certificate payload use Digital Signatures	A2		certificate payload use Digital Signatures
27				The Certificate payload **MUST** be accepted at any point during an exchange.	MUST	A2		certificate payload use Digital Signatures	A1		certificate payload use Digital Signatures

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
28	3.10	Certificate Request Payload	Certificate Request payloads	Certificate Request payloads **SHOULD** be included in an exchange whenever an appropriate directory service (e.g. Secure DNS [DNSSEC]) is not available to distribute certificates.	SHOULD	A2		certificate request payload use Digital Signatures	A1		certificate request payload use Digital Signatures
29				The Certificate Request payload **MUST** be accepted at any point during the exchange.	MUST	A2		certificate request payload use Digital Signatures	A1		certificate request payload use Digital Signatures
30				The responder to the Certificate Request payload **MUST** send its certificate, if certificates are supported, based on the values contained in the payload.	MUST	A2		certificate request payload use Digital Signatures	A2		certificate request payload use Digital Signatures
31				If multiple certificates are required, then multiple Certificate Request payloads **SHOULD** be transmitted.	SHOULD	A2		certificate request payload use Digital Signatures	A2		certificate request payload use Digital Signatures
32			certificate authority	If there is no specific certificate authority requested, this field **SHOULD** not be included.	SHOULD	A2		certificate request payload use Digital Signatures	A2		certificate request payload use Digital Signatures
33	3.14	Notification Payload	SPI Size	In the case of ISAKMP, the Initiator and Responder cookie pair from the ISAKMP Header is the ISAKMP SPI, therefore, the SPI Size is irrelevant and **MAY** be from zero (0) to sixteen (16).If the SPI Size is non-zero, the content of the SPI field **MUST** be ignored.	MAY	A2		notification payload	A2		notification payload
34					MUST	A2		notification payload	A2		notification payload
35	3.15	Delete Payload	delete multiple SPIs	It is possible to send multiple SPIs in a Delete payload, however, each SPI **MUST** be for the same protocol. Mixing of Protocol Identifiers **MUST NOT** be performed with the Delete payload.	MUST	A2		Delete Payload	A2		Delete Payload
36					MUST NOT	A2		Delete Payload	A2		Delete Payload
37	3.16	Vendor ID Payload	Vender ID	The Vendor ID payload is not an announcement from the sender that it will send private payload types. A vendor sending the Vendor ID **MUST NOT** make any assumptions about private payloads that it may send unless a Vendor ID is received as well.	MUST NOT	B		Vendor ID	B		Vendor ID
38				Multiple Vendor ID payloads **MAY** be sent.	MAY	B		Vendor ID	B		Vendor ID
39				An implementation is **NOT REQUIRED** to understand any Vendor ID payloads.	NOT REQUIRED	B		Vendor ID	B		Vendor ID
40				An implementation is **NOT REQUIRED** to send any Vendor ID payload at all.	NOT REQUIRED	B		Vendor ID	B		Vendor ID
41				If a private payload was sent without prior agreement to send it, a compliant implementation may reject a proposal with a notify message of type INVALID-PAYLOAD-TYPE.	(do)	B		Vendor ID	B		Vendor ID
42				If a Vendor ID payload is sent, it **MUST** be sent during the Phase 1 negotiation.	MUST	B		Vendor ID	B		Vendor ID
43				However, this practice **SHOULD NOT** be widespread and vendors should work towards standardization instead.	SHOULD NOT	-		not specification	-		not specification
44				The vendor defined constant **MUST** be unique.	MUST	B		Vendor ID	B		Vendor ID

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	4	ISAKMP Exchanges		This section describes the procedures for SA establishment and SA modification, followed by a default set of exchanges that **MAY** be used for initial interoperability.	MAY	-		not specification	-		not specification
2	4.1	ISAKMP Exchange Types	SA Payload	While the ordering of payloads within messages is not mandated, for processing efficiency it is **RECOMMENDED** that the Security Association payload be the first payload within an exchange.	RECOMMENDED	A1			A1		
3			DOI	The defined exchanges are not meant to satisfy all DOI and key exchange protocol requirements. If the defined exchanges meet the DOI requirements, then they can be used as outlined. If the defined exchanges do not meet the security requirements defined by the DOI, then the DOI **MUST** specify new exchange type(s) and the valid sequences of payloads that make up a successful exchange, and how to build and interpret those payloads. All ISAKMP implementations **MUST** implement the Informational Exchange and **SHOULD** implement the other four exchanges. However, this is dependent on the definition of the DOI and associated key exchange protocols.	MUST	B		IPsec DOI only	B		IPsec DOI only
4					MUST	A2		For Commit Bit and Delete payload	A2		For Commit Bit, Delete payload
5					SHOULD	B		local policy	B		local policy
6	4.1.1	Notation	Number of Proporsal and Transform payloads	SA is an SA negotiation payload with one or more Proposal and Transform payloads.	(do)	A1/A2		Phase 2 negotiation B:Initiator A:Responder	A1/A2		Phase 2 negotiation B:Initiator A:Responder
7				SA is an SA negotiation payload with one Proposal and one Transform	Add	A1			A1		
8				An initiator **MAY** provide multiple proposals for negotiation;a responder **MUST** reply with only one.	MAY	A2		multiple proposals for Initiator	A2		multiple proposals for Initiator
9					MUST	A1			A1		
10			encrypt	*' signifies payload encryption after the ISAKMP header. This encryption **MUST** begin immediately after the ISAKMP header and all payloads following the ISAKMP header **MUST** be encrypted.	MUST	A1			A1		
11					MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
12	4.2	Security Association Establishment		If the SA establishment negotiation is for a combined protection suite consisting of multiple protocols, then there **MUST** be multiple Proposal payloads each with the same Proposal number.	MUST	B		multiple protocols for Initiator	B		multiple protocols for Initiator
13				These proposals **MUST** be considered as a unit and **MUST NOT** be separated by a proposal with a different proposal number.	MUST	B		multiple protocols for Initiator	B		multiple protocols for Initiator
14					MUST NOT	B		multiple protocols for Initiator	B		multiple protocols for Initiator
15				If the SA establishment negotiation is for different protection suites, then there **MUST** be multiple Proposal payloads each with a monotonically increasing Proposal number.	MUST	B		multiple Proposal payloads for Initiator	B		multiple Proposal payloads for Initiator
16				The different proposals **MUST** be presented in the initiator's preference order.	MUST	B		multiple Proposal payloads for Initiator	B		multiple Proposal payloads for Initiator
17				The multiple transforms **MUST** be presented with monotonically increasing numbers in the initiator's preference order.	MUST	A1			A1		
18				The receiving entity **MUST** select a single transform for each protocol in a proposal or reject the entire proposal.	MUST	A1			A1		
19				When responding to a Security Association payload, the responder **MUST** send a Security Association payload with the selected proposal, which may consist of multiple Proposal payloads and their associated Transform payloads.	MUST	A1			A1		
20				Each of the Proposal payloads **MUST** contain a single Transform payload associated with the Protocol.	MUST	A1			A1		
21				The responder **SHOULD** retain the Proposal # field in the Proposal payload and the Transform # field in each Transform payload of the selected Proposal.Retention of Proposal and Transform numbers should speed the initiator's protocol processing by negating the need to compare the respondor's selection with every offered option.	SHOULD	B		local policy	B		local policy
22				The initiator **MUST** verify that the Security Association payload received from the responder matches one of the proposals sent initially.	MUST	A1			A1		
23	4.2.1	Security Association Establishment Examples		An example for this proposal might be: Protocol 1 is ESP with Transform 1 as 3DES and Transform 2 as DES AND Protocol 2 is AH with Transform 1 as SHA. The responder **MUST** select from the two transforms proposed for ESP.	MUST	A1			A1		
24				This is followed by Proposal 2 with Protocol 1 as ESP with Transform 1 as DES and Transform 2 as 3DES. The responder **MUST** select from the two different proposals. If the second Proposal is selected, the responder **MUST** select from the two transforms for ESP.	MUST	A1			A1		
25					MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
26	4.3	Security Association Modification	phase 1 negotiation	Modification of an ISAKMP SA (phase 1 negotiation) follows the same procedure as creation of an ISAKMP SA. There is no relationship between the two SAs and the initiator and responder cookie pairs **SHOULD** be different, as outlined in section 2.5.3.	SHOULD	A1			A1		
27			phase 2 negotiation	Modification of a Protocol SA (phase 2 negotiation) follows the same procedure as creation of a Protocol SA. The creation of a new SA is protected by the existing ISAKMP SA. There is no relationship between the two Protocol SAs. A protocol implementation **SHOULD** begin using the newly created SA for outbound traffic and **SHOULD** continue to support incoming traffic on the old SA until it is deleted or until traffic is received under the protection of the newly created SA.	SHOULD	A2		local policy	A2		local policy
28					SHOULD	A1			A1		
29	4.4	Base Exchange	the first message	Random information provided by both parties **SHOULD** be used by the authentication mechanism to provide shared proof of participation in the exchange.	SHOULD	B		Base Exchange	B		Base Exchange
30			the second message	Random information provided by both parties **SHOULD** be used by the authentication mechanism to provide shared proof of participation in the exchange.	SHOULD	B		Base Exchange	B		Base Exchange
31	4.5	Identity Protection Exchange	the third (3) and fourth (4) messages	Random information provided by both parties **SHOULD** be used by the authentication mechanism to provide shared proof of participation in the exchange.	SHOULD	B		Identity Protection Exchange	B		Identity Protection Exchange
32	4.6	Authentication Only Exchange	the first message	Random information provided by both parties **SHOULD** be used by the authentication mechanism to provide shared proof of participation in the exchange.	SHOULD	B		Authentication Only Exchange	B		Authentication Only Exchange
33			the second message	Random information provided by both parties **SHOULD** be used by the authentication mechanism to provide shared proof of participation in the exchange.	SHOULD	B		Authentication Only Exchange	B		Authentication Only Exchange
34	4.7	Aggressive Exchange	Identity protection	Identity protection is not provided because identities are exchanged before a common shared secret has been established and, therefore, encryption of the identities is not possible.	(do)	A2		Aggressive Exchange	A1		
35			In the first message	Random information provided by both parties **SHOULD** be used by the authentication mechanism to provide shared proof of participation in the exchange.	SHOULD	A2		Aggressive Exchange	A1		
36			In the second message	Random information provided by both parties **SHOULD** be used by the authentication mechanism to provide shared proof of participation in the exchange.	SHOULD	A2		Aggressive Exchange	A1		
37	4.8	Informational Exchange	Informational Exchange of the protection	Once keying material has been exchanged or an ISAKMP SA has been established, the Informational Exchange **MUST** be transmitted under the protection provided by the keying material or the ISAKMP SA.	MUST	B		Infomational Exchange	B		Infomational Exchange
38			cryptographic synchronization	All exchanges are similar in that with the beginning of any exchange, cryptographic synchronization **MUST** occur.Thus, the generation of an Message ID (MID) for an Informational Exchange **SHOULD** be independent of IVs of other on-going communication.When the Commit Bit is set, the Message ID field of the Informational Exchange **MUST** contain the Message ID of the original ISAKMP Phase 2 SA negotiation, rather than a new Message ID (MID).	MUST	B		Infomational Exchange	B		Infomational Exchange
39			Message ID		SHOULD	B		Infomational Exchange	B		Infomational Exchange
40			Commit Bit		MUST	A2		Commit Bit	A2		Commit Bit

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	5.1	General Message Processing	Packet length checks	All processing **SHOULD** include packet length checks to insure the packet received is at least as long as the length given in the ISAKMP Header.	SHOULD	A1			A1		
2				If the ISAKMP message length and the value in the Payload Length field of the ISAKMP Header are not the same, then the ISAKMP message **MUST** be rejected.	MUST	A1			A1		
3			Receving an ISAKMP message	The receiving entity (initiator or responder) **MUST** do the following: 1. The event, UNEQUAL PAYLOAD LENGTHS, **MAY** be logged in the appropriate system audit file. 2. An Informational Exchange with a Notification payload containing the UNEQUAL-PAYLOAD-LENGTHS message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
4					MAY	B		logging	B		logging
5					MAY	B		Infomational Exchange	B		Informational Exchange
6			Receving an ISAKMP message (Transmitting an ISAKMP message)	When transmitting an ISAKMP message, the transmitting entity (initiator or responder) **MUST** do the following: 1. Set a timer and initialize a retry counter.NOTE: Implementations **MUST NOT** use a fixed timer. Instead, transmission timer values should be adjusted dynamically based on measured round trip times. In addition, successive retransmissions of the same packet should be separated by increasingly longer time intervals (e.g., exponential backoff). 2. If the timer expires, the ISAKMP message is resent and the retry counter is decremented. 3. If the retry counter reaches zero (0), the event, RETRY LIMIT REACHED, **MAY** be logged in the appropriate system audit file. 4. The ISAKMP protocol machine clears all states and returns to IDLE.	MUST	A1			A1		
7					MUST NOT	A1			A1		
8					MUST	A1			A1		
9					MAY	B		logging	B		logging
10					MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
11	5.2	ISAKMP Header Processing	Creating an ISAKMP message	1. Create the respective cookie. See section 2.5.3 for details. 2. Determine the relevant security characteristics of the session(i.e. DOI and situation). 3. Construct an ISAKMP Header with fields as described in section3.1. 4. Construct other ISAKMP payloads, depending on the exchange type. 5. Transmit the message to the destination host as described in section5.1.	MUST	A1			A1		
12			Receving an ISAKMP message (Verify the Initiator and Responder "cookies")	1. Verify the Initiator and Responder "cookies". If the cookie validation fails, the message is discarded and the following actions are taken: (a) The event, INVALID COOKIE, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-COOKIE message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
13					MAY	B		logging	B		logging
14					MAY	B		Informational Exchange	B		Informational Exchange
15			Receving an ISAKMP message (Check the Next Payload field)	2. Check the Next Payload field to confirm it is valid. If the Next Payload field validation fails, the message is discarded and the following actions are taken: (a) The event, INVALID NEXT PAYLOAD, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-PAYLOAD-TYPE message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
16					MAY	B		logging	B		logging
17					MAY	B		Informational Exchange	B		Informational Exchange
18			Receving an ISAKMP message (Check the Major and Minor Version fields)	3. Check the Major and Minor Version fields to confirm they are correct (see section 3.1). If the Version field validation fails, the message is discarded and the following actions are taken: (a) The event, INVALID ISAKMP VERSION, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-MAJOR-VERSION or INVALID-MINOR-VERSION message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
19					MAY	B		logging	B		logging
20					MAY	B		Informational Exchange	B		Informational Exchange
21			Receving an ISAKMP message (Check the Exchange Type field)	4. Check the Exchange Type field to confirm it is valid. If the Exchange Type field validation fails, the message is discarded and the following actions are taken: (a) The event, INVALID EXCHANGE TYPE, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-EXCHANGE-TYPE message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
22					MAY	B		logging	B		logging
23					MAY	B		Informational Exchange	B		Informational Exchange

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
24			Receving an ISAKMP message (Check the Flags field)	5. Check the Flags field to ensure it contains correct values. If the Flags field validation fails, the message is discarded and the following actions are taken:	MUST	A1			A1		
25				(a) The event, INVALID FLAGS, **MAY** be logged in the appropriate systemaudit file.	MAY	B		logging	B		logging
26				(b) An Informational Exchange with a Notification payload containing the INVALID-FLAGS message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange
27			Receving an ISAKMP message (Check the Message ID)	6. Check the Message ID field to ensure it contains correct values. If the Message ID validation fails, the message is discarded and the following actions are taken:	MUST	A1			A1		
28				(a) The event, INVALID MESSAGE ID, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
29				(b) An Informational Exchange with a Notification payload containing the INVALID-MESSAGE-ID message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange
30			Receving an ISAKMP message (The Next Payload)	7. Processing of the ISAKMP message continues using the value in the Next Payload field.	MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
31	5.3	Generic Payload Header Processing	Creating the Generic Payload Header	1. Place the value of the Next Payload in the Next Payload field. These values are described in section 3.1. 2. Place the value zero (0) in the RESERVED field. 3. Place the length (in octets) of the payload in the Payload Length field. 4. Construct the payloads as defined in the remainder of this section.	MUST	A1			A1		
32			Receving the any of the ISAKMP (Check the Next Payload field)	1. Check the Next Payload field to confirm it is valid. If the Next Payload field validation fails, the message is discarded and the following actions are taken:	MUST	A1			A1		
33				(a) The event, INVALID NEXT PAYLOAD, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
34				(b) An Informational Exchange with a Notification payload containing the INVALID-PAYLOAD-TYPE message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange
35			Receving the any of the ISAKMP (Verify the RESERVED field)	2. Verify the RESERVED field contains the value zero. If the value in the RESERVED field is not zero, the message is discarded and the following actions are taken:	MUST	A1			A1		
36				(a) The event, INVALID RESERVED FIELD, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
37				(b) An Informational Exchange with a Notification payload containing the BAD-PROPOSAL-SYNTAX or PAYLOAD-MALFORMED message type **MAY** be sent to the transmitting entity. This action is dictated by a	MAY	B		Informational Exchange	B		Informational Exchange
38			Receving the any of the ISAKMP (The Next Payload)	3. Process the remaining payloads as defined by the Next Payload field.	MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
39	5.4	Security Association Payload Processing	Creating a Security Association Payload	1. Determine the Domain of Interpretation for which this negotiation is being performed. 2. Determine the situation within the determined DOI for which this negotiation is being performed. 3. Determine the proposal(s) and transform(s) within the situation. These are described, respectively, in sections 3.5 and 3.6. 4. Construct a Security Association payload. 5. Transmit the message to the receiving entity as described in section 5.1.	MUST	A1			A1		
40			Receving a Security Association Payload (checking the DOI)	1. Determine if the Domain of Interpretation (DOI) is supported. If the DOI determination fails, the message is discarded and the following actions are taken:	MUST	A1			A1		
41				(a) The event, INVALID DOI, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
42				(b) An Informational Exchange with a Notification payload containing the DOI-NOT-SUPPORTED message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		infomational	B		infomational
43			Receving a Security Association Payload (Determine if the given situation can be protected.)	2. Determine if the given situation can be protected. If the Situation determination fails, the message is discarded and the following actions are taken:	MUST	A1			A1		
44				(a) The event, INVALID SITUATION, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
45				(b) An Informational Exchange with a Notification payload containing the SITUATION-NOT-SUPPORTED message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange
46			Receving a Security Association Payload (Process the remaining payloads)	3. Process the remaining payloads (i.e. Proposal, Transform) of the Security Association Payload. If the Security Association Proposal (as described in sections 5.5 and 5.6) is not accepted, then the following actions are taken:	MUST	A1			A1		
47				(a) The event, INVALID PROPOSAL, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
48				(b) An Informational Exchange with a Notification payload containing the NO-PROPOSAL-CHOSEN message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
49	5.5	Proposal Payload Processing	Creating a Proposal Payload	1. Determine the Protocol for this proposal. 2. Determine the number of proposals to be offered for this protocol and the number of transforms for each proposal. Transforms are described in section 3.6. 3. Generate a unique pseudo-random SPI. 4. Construct a Proposal payload.	MUST	A1			A1		
50			Receving a Proposal Payload (Determine if the Protocol is supported)	1. Determine if the Protocol is supported. If the Protocol-ID field is invalid, the payload is discarded and the following actions are taken: (a) The event, INVALID PROTOCOL, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-PROTOCOL-ID message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
51					MAY	B		logging	B		logging
52					MAY	B		Informational Exchange	B		Informational Exchange
53			Receving a Proposal Payload (Determine if the SPI is valid)	2. Determine if the SPI is valid. If the SPI is invalid, the payload is discarded and the following actions are taken: (a) The event, INVALID SPI, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-SPI message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
54					MAY	B		logging	B		logging
55					MAY	B		Informational Exchange	B		Informational Exchange
56			Receving a Proposal Payload (Ensure the Proposals are formed)	3. Ensure the Proposals are presented according to the details given in section 3.5 and 4.2. If the proposals are not formed correctly, the following actions are taken: (a) Possible events, BAD PROPOSAL SYNTAX, INVALID PROPOSAL, are logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the BAD-PROPOSAL-SYNTAX or PAYLOAD-MALFORMED message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
57					MUST	B		logging	B		logging
58					MAY	B		Informational Exchange	B		Informational Exchange
59			Receving a Proposal Payload (The Next Payload)	4. Process the Proposal and Transform payloads as defined by the Next Payload field.	MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
60	5.6	Transform Payload Processing	Creating a Transform Payload	1. Determine the Transform # for this transform. 2. Determine the number of transforms to be offered for this proposal. Transforms are described in sections 3.6. 3. Construct a Transform payload.	MUST	A1			A1		
61			Receving a Transform Payload (Determine if the Transform is supported.)	1. Determine if the Transform is supported. If the Transform-ID field contains an unknown or unsupported value, then that Transform payload **MUST** be ignored and **MUST NOT** cause the generation of an INVALID TRANSFORM event. If the Transform-ID field is invalid, the payload is discarded and the following actions are taken: (a) The event, INVALID TRANSFORM, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-TRANSFORM-ID message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
62					MUST NOT	A1			A1		
63					MUST	A1			A1		
64					MAY	B		logging	B		logging
65					MAY	B		Informational Exchange	B		Informational Exchange
66			Receving a Transform Payload (Ensure the Transforms are formed)	2. Ensure the Transforms are presented according to the details given in section 3.6 and 4.2. If the transforms are not formed correctly, the following actions are taken: (a) Possible events, BAD PROPOSAL SYNTAX, INVALID TRANSFORM, INVALID ATTRIBUTES, are logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the BAD-PROPOSAL-SYNTAX, PAYLOAD-MALFORMED or ATTRIBUTES-NOT-SUPPORTED message type **MAY** be sent to the transmitting entity. This action is	MUST	A1			A1		
67					(do)	B		logging	B		logging
68					MAY	B		Informational Exchange	B		Informational Exchange
69			Receving a Transform Payload (The Next Payload)	3. Process the subsequent Transform and Proposal payloads as defined by the Next Payload field.	MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
70	5.7	Key Exchange Payload Processing	Creating a Key Exchange Payload	1. Determine the Key Exchange to be used as defined by the DOI. 2. Determine the usage of the Key Exchange Data field as defined by the DOI. 3. Construct a Key Exchange payload. 4. Transmit the message to the receiving entity as described in section	MUST	A1			A1		
71			Receving a Key Exchange payload	1. Determine if the Key Exchange is supported. If the Key Exchange determination fails, the message is discarded and the following actions are taken: (a) The event, INVALID KEY INFORMATION, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-KEY-INFORMATION message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
72					MAY	B		logging	B		logging
73					MAY	B		Informational Exchange	B		Informational Exchange
74	5.8	Identification Payload Processing	Creating an Identification Payload	1. Determine the Identification information to be used as defined by the DOI (and possibly the situation). 2. Determine the usage of the Identification Data field as defined by the DOI. 3. Construct an Identification payload. 4. Transmit the message to the receiving entity as described in section 5.1	MUST	A1			A1		
75			Receving an Identification Payload	1. Determine if the Identification Type is supported. This may be based on the DOI and Situation. If the Identification determination fails, the message is discarded and the following actions are taken: (a) The event, INVALID ID INFORMATION, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-ID-INFORMATION message type **MAY** be sent to the transmitting entity. This action is dictated by a	MUST	A1			A1		
76					MAY	B		logging	B		logging
77					MAY	B		Informational Exchange	B		Informational Exchange

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
78	5.9	Certificate Payload Processing	Creating a Certificate Payload	1. Determine the Certificate Encoding to be used. This may be specified by the DOI. 2. Ensure the existence of a certificate formatted as defined by the Certificate Encoding. 3. Construct a Certificate payload. 4. Transmit the message to the receiving entity as described in section	MUST	A1			A1		
79			Receving a Certificate Payload(Determine if the Certificate Encoding is supported)	1. Determine if the Certificate Encoding is supported. If the Certificate Encoding is not supported, the payload is discarded and the following actions are taken: (a) The event, INVALID CERTIFICATE TYPE, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-CERT-ENCODING message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
80					MAY	B		logging	B		logging
81					MAY	B		Informational Exchange	B		Informational Exchange
82			Receving a Certificate Payload(Process the Certificate Data field)	2. Process the Certificate Data field. If the Certificate Data is invalid or improperly formatted, the payload is discarded and the following actions are taken: (a) The event, INVALID CERTIFICATE, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-CERTIFICATE message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
83					MAY	B		logging	B		logging
84					MAY	B		Informational Exchange	B		Informational Exchange

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
85	5.10	Certificate Request Payload Processing	Creating a Certificate Request Payload	1. Determine the type of Certificate Encoding to be requested. This may be specified by the DOI. 2. Determine the name of an acceptable Certificate Authority which is to be requested (if applicable). 3. Construct a Certificate Request payload. 4. Transmit the message to the receiving entity as described in section 5.1.	MUST	A1			A1		
86			Receving a Certificate Request Payload(Determine if the Certificate Encoding is supported)	1. Determine if the Certificate Encoding is supported. If the Certificate Encoding is invalid, the payload is discarded and the following actions are taken:	MUST	A1			A1		
87				(a) The event, INVALID CERTIFICATE TYPE, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
88				(b) An Informational Exchange with a Notification payload containing the INVALID-CERT-ENCODING message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange
				If the Certificate Encoding is not supported, the payload is discarded and the following actions are taken: (a) The event, CERTIFICATE TYPE UNSUPPORTED, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
89				(b) An Informational Exchange with a Notification payload containing the CERT-TYPE-UNSUPPORTED message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange
90			Receving a Certificate Request Payload(Determine if the Certificate Authority is supported)	2. Determine if the Certificate Authority is supported for the specified Certificate Encoding. If the Certificate Authority is invalid or improperly formatted, the payload is discarded and the following actions are taken:	MUST	A1			A1		
91				(a) The event, INVALID CERTIFICATE AUTHORITY, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
92				(b) An Informational Exchange with a Notification payload containing the INVALID-CERT-AUTHORITY message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange
93			Receving a Certificate Request Payload(Process the Certificate Request)	3. Process the Certificate Request. If a requested Certificate Type with the specified Certificate Authority is not available, then the payload is discarded and the following actions are taken:	MUST	A1			A1		
94				(a) The event, CERTIFICATE-UNAVAILABLE, **MAY** be logged in the appropriate system audit file.	MAY	B		logging	B		logging
95				(b) An Informational Exchange with a Notification payload containing the CERTIFICATE-UNAVAILABLE message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MAY	B		Informational Exchange	B		Informational Exchange

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
96	5.11	Hash Payload Processing	Creating a Hash Payload	1. Determine the Hash function to be used as defined by the SA negotiation. 2. Determine the usage of the Hash Data field as defined by the DOI. 3. Construct a Hash payload. 4. Transmit the message to the receiving entity as described in section 5.1.	MUST	A1			A1		
97			Receving a Hash Payload (Determine if the Hash is supported.)	1. Determine if the Hash is supported. If the Hash determination fails, the message is discarded and the following actions are taken: (a) The event, INVALID HASH INFORMATION, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-HASH-INFORMATION message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
98					MAY	B		logging	B		logging
99					MAY	B		Informational Exchange	B		Informational Exchange
100			Receving a Hash Payload (Perform the Hash function)	2. Perform the Hash function as outlined in the DOI and/or Key Exchange protocol documents. If the Hash function fails, the message is discarded and the following actions are taken: (a) The event, INVALID HASH VALUE, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the AUTHENTICATION-FAILED message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
101					MAY	B		logging	B		logging
102					MAY	B		Informational Exchange	B		Informational Exchange
103	5.12	Signature Payload Processing	Creating a Signature Payload Processing	1. Determine the Signature function to be used as defined by the SA negotiation. 2. Determine the usage of the Signature Data field as defined by the DOI. 3. Construct a Signature payload. 4. Transmit the message to the	MUST	A1			A1		
104			Receiving a Signature Payload Processing(Determine if the Signature is supported)	1. Determine if the Signature is supported. If the Signature determination fails, the message is discarded and the following actions are taken: (a) The event, INVALID SIGNATURE INFORMATION, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the INVALID-SIGNATURE message type **MAY** be sent to the transmitting entity. This action is dictated by a	MUST	A1			A1		
105					MAY	B		logging	B		logging
106					MAY	B		Informational Exchange	B		Informational Exchange
107			Receiving a Signature Payload Processing(Perform the Signature function)	2. Perform the Signature function as outlined in the DOI and/or Key Exchange protocol documents. If the Signature function fails, the message is discarded and the following actions are taken: (a) The event, INVALID SIGNATURE VALUE, **MAY** be logged in the appropriate system audit file. (b) An Informational Exchange with a Notification payload containing the AUTHENTICATION-FAILED message type **MAY** be sent to the transmitting entity. This action is dictated by a system security policy.	MUST	A1			A1		
108					MAY	B		logging	B		logging
109					MAY	B		Informational Exchange	B		Informational Exchange

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
110	5.13	Nonce Payload Processing	Creating a Nonce Payload	1. Create a unique random value to be used as a nonce. 2. Construct a Nonce payload. 3. Transmit the message to the receiving entity as described in section 5.1.	MUST	A1			A1		
111			Receving a Nonce Payload	1. There are no specific procedures for handling Nonce payloads. The procedures are defined by the exchange types (and possibly the DOI and Key Exchange descriptions).	MUST	A1			A1		
112	5.14	Notification Payload Processing		The Informational Exchange with a Notify Payload provides a controlled method of informing a peer entity that errors have occurred during protocol processing. It is **RECOMMENDED** that Notify Payloads be sent in a separate Informational Exchange rather than appending a Notify Payload to an existing exchange.	RECOMMENDED	B		Notification Payload	B		Notification Payload
113			Creating a Notification Payload	1. Determine the DOI for this Notification. 2. Determine the Protocol-ID for this Notification. 3. Determine the SPI size based on the Protocol-ID field. This field is necessary because different security protocols have different SPI sizes. For example, ISAKMP combines the Initiator and Responder cookie pair (16 octets) as a SPI, while ESP and AH have 4 octet SPIs. 4. Determine the Notify Message Type based on the error or status message desired. 5. Determine the SPI which is associated with this notification. 6. Determine if additional Notification Data is to be included. This is additional information specified by the DOI. 7. Construct a Notification payload. 8. Transmit the message to the	MUST	A1			A1		
114			a NOTIFICATION PAYLOAD ERROR event	Because the Informational Exchange with a Notification payload is a unidirectional message a retransmission will not be performed. The local security policy will dictate the procedures for continuing. However, we **RECOMMEND** that a NOTIFICATION PAYLOAD ERROR event be logged in the appropriate system audit file by the receiving	RECOMMEND	B		Notification Payload	B		Notification Payload
115			the protection provided by the keying material or the ISAKMP SA	Once the keying material has been exchanged or the ISAKMP SA has been established, the Informational Exchange **MUST** be transmitted under the protection provided by the keying material or the ISAKMP SA.	MUST	B		Notification Payload	B		Notification Payload
116			Receiving a Notification Payload(Determine if the Informational Exchange has any protection applied to it)	1. Determine if the Informational Exchange has any protection applied to it by checking the Encryption Bit and the Authentication Only Bit in the ISAKMP Header. If the Encryption Bit is set, i.e. the Informational Exchange is encrypted, then the message **MUST** be decrypted using the (in-progress or completed) ISAKMP SA. Once the decryption is complete the processing can continue as described below. If the Authentication Only Bit is set, then the message **MUST** be authenticated using the (in-progress or completed) ISAKMP SA. Once the authentication is completed, the processing can continue as described below. If the Informational Exchange is not encrypted or authentication, the payload processing can continue as described below.	MUST	B		Notification Payload	B		Notification Payload
117					MUST	B		Notification Payload	B		Notification Payload
118					MUST	B		Notification Payload	B		Notification Payload

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
119			Receiving a Notification Payload(Determine if the Domain of Interpretation (DOI) is supported)	2. Determine if the Domain of Interpretation (DOI) is supported. If the DOI determination fails, the payload is discarded and the following action is taken: (a) The event, INVALID DOI, MAY be logged in the appropriate system audit file.	MUST	B		Notification Payload	B		Notification Payload
120					MAY	B		Notification Payload	B		Notification Payload
121			Receiving a Notification Payload(Determine if the Protocol-Id is supported)	3. Determine if the Protocol-Id is supported. If the Protocol-Id determination fails, the payload is discarded and the following action is taken: (a) The event, INVALID PROTOCOL-ID, **MAY** be logged in the appropriate system audit file.	MUST	B		Notification Payload	B		Notification Payload
122					MAY	B		Notification Payload	B		Notification Payload
123			Receiving a Notification Payload(Determine if the SPI is valid)	4. Determine if the SPI is valid. If the SPI is invalid, the payload is discarded and the following action is taken: (a) The event, INVALID SPI, **MAY** be logged in the appropriate system audit file.	MUST	B		Notification Payload	B		Notification Payload
124					MAY	B		Notification Payload	B		Notification Payload
125			Receiving a Notification Payload(Determine if the Notify Message Type is valid)	5. Determine if the Notify Message Type is valid. If the Notify Message Type is invalid, the payload is discarded and the following action is taken: (a) The event, INVALID MESSAGE TYPE, **MAY** be logged in the appropriate system audit file.	MUST	B		Notification Payload	B		Notification Payload
126					MAY	B		Notification Payload	B		Notification Payload
127			Receiving a Notification Payload(Process the Notification payload, including additional Notification	6. Process the Notification payload, including additional Notification Data, and take appropriate action, according to local security policy.	MUST	B		Notification Payload	B		Notification Payload
128	5.15	Delete Payload Processing	Creating a Delete Payload	1. Determine the DOI for this Deletion. 2. Determine the Protocol-ID for this Deletion. 3. Determine the SPI size based on the Protocol-ID field. This field is necessary because different security protocols have different SPI sizes. For example, ISAKMP combines the Initiator and Responder cookie pair (16 octets) as a SPI, while ESP and AH have 4 octet SPIs. 4. Determine the # of SPIs to be deleted for this protocol. 5. Determine the SPI(s) which is (are) associated with this deletion. 6. Construct a Delete payload. 7. Transmit the message to the receiving entity as described in section	MUST	A2		Delete Payload	A2		Delete Payload
129			a DELETE PAYLOAD ERROR event	Because the Informational Exchange with a Delete payload is a unidirectional message a retransmission will not be performed. The local security policy will dictate the procedures for continuing. However, we **RECOMMEND** that a DELETE PAYLOAD ERROR event be logged in the appropriate system audit file by the receiving entity.	RECOMMEND	B		Delete Payload	B		Delete Payload
130			the protection provided by an ISAKMP SA	As described above, the Informational Exchange with a Delete payload **MUST** be transmitted under the protection provided by an ISAKMP SA.	MUST	A2		Delete Payload	A2		Delete Payload

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
131			Receiving a Delete Payload(Because the Informational Exchange is protected by some security service)	1. Because the Informational Exchange is protected by some security service (e.g. authentication for an Auth-Only SA, encryption for other exchanges), the message **MUST** have these security services applied using the ISAKMP SA. Once the security service processing is complete the processing can continue as described below. Any errors that occur during the security service processing will be evident when checking information in the Delete payload. The local security policy **SHOULD** dictate any action to be taken as a result of security service processing errors.	MUST	A2		Delete Payload	A2		Delete Payload
132					MUST	A2		Delete Payload	A2		Delete Payload
133					SHOULD	B		local policy	B		local policy
134			Receiving a Delete Payload(Determine if the Domain of Interpretation (DOI) is supported)	2. Determine if the Domain of Interpretation (DOI) is supported. If the DOI determination fails, the payload is discarded and the following action is taken: (a) The event, INVALID DOI, **MAY** be logged in the appropriate system audit file.	MUST	A2		Delete Payload	A2		Delete Payload
135					MAY	B		logging	B		logging
136			Receiving a Delete Payload(Determine if the Protocol-Id is supported)	3. Determine if the Protocol-Id is supported. If the Protocol-Id determination fails, the payload is discarded and the following action is taken: (a) The event, INVALID PROTOCOL-ID, **MAY** be logged in the appropriate system audit file.	MUST	A2		Delete Payload	A2		Delete Payload
137					MAY	B		logging	B		logging
138			Receiving a Delete Payload(Determine if the SPI is valid for each SPI included in the Delete payload)	4. Determine if the SPI is valid for each SPI included in the Delete payload. For each SPI that is invalid, the following action is taken: (a) The event, INVALID SPI, **MAY** be logged in the appropriate system audit file.	MUST	A2		Delete Payload	A2		Delete Payload
139					MAY	B		logging	B		logging
140			Receiving a Delete Payload(Process the Delete payload and take appropriate action)	5. Process the Delete payload and take appropriate action, according to local security policy. As described above, one appropriate action **SHOULD** include cleaning up the local SA database.	SHOULD	A2		Delete Payload	A2		Delete Payload

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	3.2	Notation	multiple proposals	SA is an SA negotiation payload with one or more proposals. An initiator **MAY** provide multiple proposals for negotiation; a responder **MUST** reply with only one.	MAY	B		multiple proposals	B		multiple proposals
2					MUST	A1			A1		
3			encryption	Message encryption (when noted by a '*' after the ISAKMP header) **MUST** begin immediately after the ISAKMP header. When communication is protected, all payloads following the ISAKMP header **MUST** be encrypted.	MUST	A1			A1		
4					MUST	A1			A1		
5	3.3	Perfect Forward Secrecy	PFS	For PFS to exist the key used to protect transmission of data **MUST NOT** be used to derive any additional keys, and if the key used to protect transmission of data was derived from some other keying material, that material **MUST NOT** be used to derive any more keys.	MUST NOT	A2		PFS	A2		PFS
6					MUST NOT	A2		PFS	A2		PFS

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	4	Introduction	phase 1	"Main Mode" and "Aggressive Mode" each accomplish a phase 1 exchange. "Main Mode" and "Aggressive Mode" **MUST** ONLY be used in phase 1.	MUST	A1/A2		A1: Main Mode A2: Aggressive Mode	A1/A2		A1: Aggressive Mode A2: Main Mode
2			phase 2	"Quick Mode" accomplishes a phase 2 exchange. "Quick Mode" **MUST** ONLY be used in phase 2.	MUST	A1			A1		
3			New Group Mode	"New Group Mode" **MUST** ONLY be used after phase 1.	MUST	B		New Group Mode	B		New Group Mode
4			cookies	In other words, the cookies **MUST NOT** swap places when the direction of the ISAKMP SA changes.	MUST NOT	A1			A1		
5			DOI and situation	The ISAKMP SA, established in phase 1, **MAY** use the DOI and situation from a non- ISAKMP service (such as the IETF IPSec DOI [Pip97]). In this case an implementation **MAY** choose to restrict use of the ISAKMP SA for establishment of SAs for services of the same DOI. Alternately, an ISAKMP SA **MAY** be established with the value zero in both the DOI and situation (see [MSST98] for a description of these fields) and in this case implementations will be free to establish security services for any defined DOI using this ISAKMP SA.	MAY	B		the DOI and situation from a non- ISAKMP service	B		the DOI and situation from a non- ISAKMP service
6					MAY	B		the DOI and situation from a non- ISAKMP service	B		the DOI and situation from a non- ISAKMP service
7					MAY	B		the DOI and situation from a non- ISAKMP service	B		the DOI and situation from a non- ISAKMP service
8			attributes are mandatory	The following attributes are used by IKE and are negotiated as part of the ISAKMP Security Association. (These attributes pertain only to the ISAKMP Security Association and not to any Security Associations that ISAKMP may be negotiating on behalf of other services.) - encryption algorithm - hash algorithm - authentication method - information about a group over which to do Diffie-Hellman. All of these attributes are mandatory and **MUST** be negotiated.	MUST	A1			A1		
9			hash algorithm	The selected hash algorithm **MUST** support both native and HMAC modes.	MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
10			Diffie-Hellman group	The Diffie-Hellman group **MUST** be either specified using a defined group description (section 6) or by defining all attributes of a group (section 5.6).Group attributes (such as group type or prime-- see Appendix A) **MUST NOT** be offered in conjunction with a previously defined group (either a reserved group description or a private use description that is established after conclusion of a New Group Mode exchange).	MUST	A1/A2/B		A1: MODP group number 2 A2: MODP group number 1, 5, 14 B: defining all attributes of a group (section 5.6 New Group)	A1/A2/B		A1: MODP group number 2 A2: MODP group number 1, 5, 14 B: defining all attributes of a group (section 5.6 New Group)
11					MUST NOT	B		New Group Mode	B		New Group Mode
12			support the attribute values	IKE implementations **MUST** support the following attribute values: - DES [DES] in CBC mode with a weak, and semi-weak, key check (weak and semi-weak keys are referenced in [Sch96] and listed in Appendix A). The key is derived according to Appendix B. - MD5 [MD5] and SHA [SHA}. - Authentication via pre-shared keys. - MODP over default group number one (see below).	MUST	A1/A2		A: SHA pre-shared keys B: DES MD5 MODP over default group number one	A1/A2		A: SHA pre-shared keys B: DES MD5 MODP over default group number one
13				In addition, IKE implementations **SHOULD** support: 3DES for encryption; Tiger ([TIGER]) for hash; the Digital Signature Standard, RSA [RSA] signatures and authentication with RSA public key encryption; and MODP group number 2.	SHOULD	A1/A2/B		A1: 3DES for encryption MODP group number 2 A2: RSA signatures B: Tiger for hash Digital Signature Standard authentication with RSA public key encryption	A1/A2/B		A1: 3DES for encryption MODP group number 2 A2: RSA signatures B: Tiger for hash Digital Signature Standard authentication with RSA public key encryption
14			additional encryption algorithms	IKE implementations **MAY** support any additional encryption algorithms defined in Appendix A and **MAY** support ECP and EC2N groups.	MAY	B		other encryption algorithms	B		other encryption algorithms
15					MAY	B		ECP and EC2N groups	B		ECP and EC2N groups
16			DOI	The IKE modes described here **MUST** be implemented whenever the IETF IPsec DOI [Pip97] is implemented. Other DOIs **MAY** use the modes described here.	MUST	A1			A1		
17					MAY	B		Other DOIs	B		Other DOIs

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	5	Exchanges	Main Mode	Main Mode **MUST** be implemented; Aggressive Mode **SHOULD** be implemented. In addition, Quick Mode **MUST** be implemented as a mechanism to generate fresh keying material and negotiate non-ISAKMP security services.	MUST	A1			A2/B		B: Main Mode with a Pre-Shared Key A2: Main Mode with a RSS signatures
2			Aggressive Mode		SHOULD	A2		Aggressive mode	A1		A: Aggressive Mode with a Pre-Shared Key
3			Quick Mode		MUST	A1			A1		
4			New Group Mode	In addition, New Group Mode **SHOULD** be implemented as a mechanism to define private groups for Diffie-Hellman exchanges.	SHOULD	B		New Group Mode	B		New Group Mode
5			exchange type in the middle of an exchange.	Implementation **MUST NOT** switch exchange types in the middle of an exchange.	MUST NOT	A1			A1		
6			SA payload	The SA payload **MUST** precede all other payloads in a phase 1 exchange.	MUST	A1			A1		
7			the length of Diffie-Hellman public value	The Diffie-Hellman public value passed in a KE payload, in either a phase 1 or phase 2 exchange, **MUST** be the length of the negotiated Diffie-Hellman group enforced, if necessary, by pre-pending the value with zeros.	MUST	A1			A1		
8			the length of nonce payload	The length of nonce payload **MUST** be between 8 and 256 bytes inclusive.	MUST	A1			A1		
9			Aggressive Mode	The final message **MAY NOT** be sent under protection of the ISAKMP SA allowing each party to postpone exponentiation, if desired, until negotiation of this exchange is complete.	MAY NOT	A2/B		A2: If aggressive mode support, responder support the final message with protection of the ISAKMP SA and without protection of the ISAKMP SA B: Initiator	A1/B		A1: Responder support the final message with protection of the ISAKMP SA and without protection of the ISAKMP SA B: Initiator
10			a Certificate Request payload	Receipt of a Certificate Request payload **MUST NOT** extend the number of messages transmitted or expected.	MUST NOT	A2		Certificate Request payload	A2		Certificate Request payload
11			phase 1 exchanges	If multiple offers are being made for phase 1 exchanges (Main Mode and Aggressive Mode) they **MUST** take the form of multiple Transform Payloads for a single Proposal Payload in a single SA payload.To put it another way, for phase 1 exchanges there **MUST NOT** be multiple Proposal Payloads for a single SA payload and there **MUST NOT** be multiple SA payloads.	MUST	A1/A2		A1: Responder Process multiple Transform Payloads A2: Initiator transmit multiple Transform Payloads	A1/A2		A: Responder Process multiple Transform Payloads B: Initiator transmit multiple Transform Payloads
12					MUST NOT	A1			A1		
13					MUST NOT	A1			A1		
14			limit the number of offers	There is no limit on the number of offers the initiator may send to the responder but conformant implementations **MAY** choose to limit the number of offers it will inspect for performance reasons.	MAY	B		This function is implementaion-dependent.	B		This function is implementaion-dependent.
15			attributes	Responders **MUST NOT** modify attributes of any offer, attribute encoding excepted (see Appendix A). *a extract Appendix A Attributes described as basic MUST NOT be encoded as variable. Variable length attributes MAY be encoded as basic attributes if their value can fit into two octets. If this is the case, an attribute offered as variable (or basic) by the initiator of this protocol MAY be returned to the initiator as a basic (or variable).	MUST NOT	A1			A1		
16				If the initiator of an exchange notices that attribute values have changed or attributes have been added or deleted from an offer made, that response **MUST** be rejected.	MUST	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
17	5.1	IKE Phase 1 Authenticated With Signatures		In addition, there is no binding between the OIDs used for RSA signatures in PKCS #1 and those used in this document. Therefore, RSA signatures **MUST** be encoded as a private key encryption in PKCS #1 format and not as a signature in PKCS #1 format (which includes the OID of the hash algorithm). DSS signatures **MUST** be encoded as r followed by s.	MUST	A2		RSS signatures	A2		RSS signatures
18					MUST	B		DSS signatures	B		DSS signatures
19				One or more certificate payloads **MAY** be optionally passed.	MAY	A2		multiple certificate payload	A2		multiple certificate payload
20	5.2	Phase 1 Authenticated With Public Key Encryption	Public Key Encryption	If the authentication method is public key encryption, the nonce and identity payloads **MUST** be encrypted with the public key of the other party.	MUST	A2		public key encryption	A2		Public Key Encryption
21				RSA encryption **MUST** be encoded in PKCS #1 format.	MUST	A2		public key encryption	A2		Public Key Encryption
22	5.3	Phase 1 Authenticated With a Revised Mode of Public Key Encryption	A Revised Mode of Public Key Encryption	If the HASH payload is sent it **MUST** be the first payload of the second message exchange and **MUST** be followed by the encrypted nonce. If the HASH payload is not sent, the first payload of the second message exchange **MUST** be the encrypted nonce.	MUST	B		A Revised Mode of Public Key Encryption	B		A Revised Mode of Public Key Encryption
23					MUST	B		A Revised Mode of Public Key Encryption	B		A Revised Mode of Public Key Encryption
24					MUST	B		A Revised Mode of Public Key Encryption	B		A Revised Mode of Public Key Encryption
25				For brevity, only derivation of Ke_i is shown; Ke_r is identical. The length of the value 0 in the computation of K1 is a single octet. Note that Ne_i, Ne_r, Ke_i, and Ke_r are all ephemeral and **MUST** be discarded after use.	MUST	B		A Revised Mode of Public Key Encryption	B		A Revised Mode of Public Key Encryption
26				All payloads-- in whatever order-- following the encrypted nonce **MUST** be encrypted with Ke_i or Ke_r depending on the direction.	MUST	B		A Revised Mode of Public Key Encryption	B		A Revised Mode of Public Key Encryption
27	5.4	Authentication with a Pre-Shared Key	using pre-shared key authentication with Main Mode	When using pre-shared key authentication with Main Mode the key can only be identified by the IP address of the peers since HASH_I must be computed before the initiator has processed IDir.	(do)	A1			B		pre-shared key authentication with Main Mode
28			Aggressive Mode	Aggressive Mode allows for a wider range of identifiers of the pre-shared secret to be used. In addition, Aggressive Mode allows two parties to maintain multiple, different pre-shared keys and identify the correct one for a particular exchange.	(do)	A2		Aggressive mode with a Pre-Shared Key	A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
29	5.5	Phase 2 - Quick Mode		The information exchanged along with Quickv Mode **MUST** be protected by the ISAKMP SA-- i.e. all payloads except the ISAKMP header are encrypted. In Quick Mode, a HASH payload **MUST** immediately follow the ISAKMP header and a SA payload **MUST** immediately follow the HASH.	MUST	A1			A1		
30					MUST	A1			A1		
31					MUST	A1			A1		
32				While use of the key exchange payload with Quick Mode is optional it **MUST** be supported.	MUST	A2		PFS	A2		PFS
33				If ISAKMP is acting as a client negotiator on behalf of another party, the identities of the parties **MUST** be passed as IDci and then IDcr. Local policy will dictate whether the proposals are acceptable for the identities specified. If the client identities are not acceptable to the Quick Mode responder (due to policy or other reasons), a Notify payload with Notify Message Type INVALID-ID-INFORMATION (18) **SHOULD** be sent.	MUST	A1			A1		
34					SHOULD	B		Notification Payload	B		Notification Payload
35				All offers made during a Quick Mode are logically related and must be consistant. For example, if a KE payload is sent, the attribute describing the Diffie-Hellman group (see section 6.1 and [Pip97]) **MUST** be included in every transform of every proposal of every SA being negotiated. Similarly, if client identities are used, they **MUST** apply to every SA in the negotiation.	MUST	A2		PFS	A2		PFS
36					MUST	A1			A1		
37				This keying material (whether with PFS or without, and whether derived directly or through concatenation) **MUST** be used with the negotiated SA.	MUST	A1			A1		
38	5.6	New Group Mode	New Group Mode	New Group Mode **MUST NOT** be used prior to establishment of an ISAKMP SA.	MUST NOT	B		New Group Mode	B		New Group Mode
39				The description of a new group **MUST** only follow phase 1 negotiation. (It is not a phase 2 exchange, though).	MUST	B		New Group Mode	B		New Group Mode
40				The proposal will specify the characteristics of the group (see appendix A, "Attribute Assigned Numbers"). Group descriptions for private Groups **MUST** be greater than or equal to 2^15.	MUST	B		New Group Mode	B		New Group Mode
41				If the group is not acceptable, the responder **MUST** reply with a Notify payload with the message type set to ATTRIBUTES-NOT-SUPPORTED (13).	MUST	B		New Group Mode	B		New Group Mode
42				ISAKMP implementations **MAY** require private groups to expire with the SA under which they were established.	MAY	B		New Group Mode	B		New Group Mode
43	5.7	ISAKMP Informational Exchanges	Informational Exchanges	As noted the message ID in the ISAKMP header-- and used in the prf computation-- is unique to this exchange and **MUST NOT** be the same as the message ID of another phase 2 exchange which generated this informational exchange.	MUST NOT	B		Informational Exchange	B		Informational Exchange

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	6.1	First Oakley Default Group		Oakley implementations **MUST** support a MODP group with the following prime and generator. This group is assigned id 1 (one). The prime is: 2^768 - 2 ^704 - 1 + 2^64 * { [2^638 pi] + 149686 } Its hexadecimal value is FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1 29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245 E485B576 625E7EC6 F44C42E9 A63A3620 FFFFFFFF FFFFFFFF The generator is: 2.	MUST	A2		MODP group number 1	A2		MODP group number 1
2	6.2	Second Oakley Group		IKE implementations **SHOULD** support a MODP group with the following prime and generator. This group is assigned id 2 (two). The prime is 2^1024 - 2^960 - 1 + 2^64 * { [2^894 pi] + 129093 }. Its hexadecimal value is FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1 29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245 E485B576 625E7EC6 F44C42E9 A637ED6B 0BFF5CB6 F406B7ED EE386BFB 5A899FA5 AE9F2411 7C4B1FE6 49286651 ECE65381 FFFFFFFF FFFFFFFF The generator is 2 (decimal)	SHOULD	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
3	6.3	Third Oakley Group		IKE implementations **SHOULD** support a EC2N group with the following characteristics. This group is assigned id 3 (three). The curve is based on the Galois Field GF[2^155]. The field size is 155. The irreducible polynomial for the field is: u^155 + u^62 + 1. The equation for the elliptic curve is: y^2 + xy = x^3 + ax^2 + b. Field Size: 155 Group Prime/Irreducible Polynomial: 0x0800000000000000000000004000000000000001 Group Generator One: 0x7b Group Curve A: 0x0 Group Curve B: 0x07338f Group Order: 0X0800000000000000000057db5698537193aef944 The data in the KE payload when using this group is the value x from the solution (x,y), the point on the curve chosen by taking the randomly chosen secret Ka and computing Ka*P, where * is the repetition of the group addition and double operations, P is the curve point with x coordinate equal to generator 1 and the y coordinate determined from the defining equation. The equation of curve is implicitly known by the Group Type and the A and B coefficients. There are two possible values for the y coordinate; eith	SHOULD	B		MODP group number 3	B		MODP group number 3
4	6.4	Fourth Oakley Group		IKE implementations **SHOULD** support a EC2N group with the following characteristics. This group is assigned id 4 (four). The curve is based on the Galois Field GF[2^185]. The field size is 185. The irreducible polynomial for the field is: u^185 + u^69 + 1. The equation for the elliptic curve is: y^2 + xy = x^3 + ax^2 + b. Field Size: 185 Group Prime/Irreducible Polynomial: 0x020000000000000000000000000000200000000000000001 Group Generator One: 0x18 Group Curve A: 0x0 Group Curve B: 0x1ee9 Group Order: 0X01ffffffffffffffffffffffdbf2f889b73e484175f94ebc The data in the KE payload when using this group will be identical to that as when using Oakley Group 3 (three).	SHOULD	B		MODP group number 4	B		MODP group number 4

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	7.1	Phase 1 using Main Mode(Aggressive Mode)	Phase 1	The initiator **MAY** propose several proposals; the responder **MUST** reply with one.	MAY	B		several proposals	B		several proposals
					MUST	A1			A1		
2	7.2	Phase 2 with Quick Mode	Phase 2	The initiator **MAY** propose several proposals; the responder **MUST** reply with one.	MAY	B		several proposals	B		several proposals
3					MUST	A1			A1		
4			the third message	As a check against replay attacks, the responder waits until receipt of the next message.	(do)	A1			A1		

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	8	Perfect Forward Secrecy Example	PFS of both keys and all identities	To provide Perfect Forward Secrecy of both keys and all identities, two parties would perform the following: o A Main Mode Exchange to protect the identities of the ISAKMP peers. This establishes an ISAKMP SA. o A Quick Mode Exchange to negotiate other security protocol protection. This establishes a SA on each end for this protocol. o Delete the ISAKMP SA and its associated state.	(do)	B		PFS of both keys and all identities	B		PFS of both keys and all identities

No.	RFC Section	RFC Section title	Item	Functional Specification	RFC Status	IKE			IKE for MIPv6		
						Test Priority	Test No.	Reason of TEST Priority	Test Priority	Test No.	Reason of TEST Priority
1	9	Implementation Hints	no PFS	As long as the Phase 1 state remains cached, and PFS is not needed, Phase 2 can proceed without any exponentiation.	(do)	A1			A1		
2			rekeying	When one peer feels it is time to change SAs they simply use the next one within the stated range.	(do)	A1			A1		
3			Quick Mode	A range of SAs can be established by negotiating multiple SAs (identical attributes, different SPIs) with one Quick Mode.	(do)	B		multiple SAs	B		multiple SAs
4			teme for establishing Sscurity Associations	An optimization that is often useful is to establish Security Associations with peers before they are needed so that when they become needed they are already in place.	(do)	A1			A1		
5			Don't respond to any Informational exchanges	It is strongly suggested that these Informational exchanges not be responded to under any circumstances.	(do)	B		Informational exchanges	B		Informational exchanges