

**3GPP2 C.S0067-A**

**Version 1.0**

**Date: February 2009**



**3RD GENERATION  
PARTNERSHIP  
PROJECT 2  
"3GPP2"**

---

## ***Key Exchange Protocols for cdma2000 High Rate Packet Data Air Interface***

### ***COPYRIGHT***

3GPP2 and its Organizational Partners claim copyright in this document and individual Organizational Partners may copyright and issue documents or standards publications in individual Organizational Partner's name based on this document. Requests for reproduction of this document should be directed to the 3GPP2 Secretariat at [secretariat@3gpp2.org](mailto:secretariat@3gpp2.org). Requests to reproduce individual Organizational Partner's documents should be directed to that Organizational Partner. See [www.3gpp2.org](http://www.3gpp2.org) for more information.

No text.

1 FOREWORD..... ix

2 REFERENCES..... xi

3 Normative References ..... xi

4 Informative References ..... xi

5 1 Generic Key Exchange Protocol ..... 1-1

6 1.1 Overview..... 1-1

7 1.2 Primitives and Public Data ..... 1-1

8 1.2.1 Commands ..... 1-1

9 1.2.2 Return Indications..... 1-1

10 1.2.3 Public Data..... 1-1

11 1.2.4 Interface to Other Protocols ..... 1-2

12 1.2.4.1 Commands ..... 1-2

13 1.2.4.2 Indications..... 1-2

14 1.3 Protocol Data Unit ..... 1-2

15 1.4 Protocol Initialization for the InConfiguration Protocol Instance..... 1-2

16 1.5 Procedures and Messages for the InConfiguration Instance of the Protocol..... 1-3

17 1.5.1 Procedures ..... 1-3

18 1.5.2 Commit Procedures ..... 1-3

19 1.5.3 Message Formats..... 1-4

20 1.5.3.1 ConfigurationRequest ..... 1-4

21 1.5.3.2 ConfigurationResponse ..... 1-4

22 1.6 Procedures and Messages for the InUse Instance of the Protocol ..... 1-5

23 1.6.1 Procedures ..... 1-5

24 1.6.1.1 Access Terminal Requirements ..... 1-5

25 1.6.1.1.1 Processing a KeyRequest message..... 1-5

26 1.6.1.1.2 Processing the ANKeyComplete message ..... 1-6

27 1.6.1.2 Access Network Requirements ..... 1-6

28 1.6.1.3 Session Key Derivation ..... 1-7

29 1.6.1.4 MICKey, Authentication Key and Encryption Key Generation..... 1-7

30 1.6.2 Message Formats..... 1-8

31 1.6.2.1 KeyRequest..... 1-8

32 1.6.2.2 KeyResponse..... 1-9

1	1.6.2.3 ANKeyComplete .....	1-11
2	1.6.2.4 AttributeUpdateRequest .....	1-12
3	1.6.2.5 AttributeUpdateAccept .....	1-13
4	1.6.2.6 AttributeUpdateReject .....	1-13
5	1.6.3 Interface to Other Protocols .....	1-14
6	1.6.3.1 Commands .....	1-14
7	1.6.3.2 Indications .....	1-14
8	1.7 Configuration Attributes .....	1-14
9	1.8 Protocol Numeric Constants .....	1-15
10	1.9 Session State Information .....	1-15
11	1.9.1 SKey Parameter .....	1-15
12	1.9.2 PMK Parameter .....	1-17
13	2 Mutli-Key Key Exchange Protocol.....	2-1
14	2.1 Overview .....	2-1
15	2.2 Primitives and Public Data .....	2-1
16	2.2.1 Commands.....	2-1
17	2.2.2 Return Indications .....	2-1
18	2.2.3 Public Data .....	2-1
19	2.2.4 Interface to Other Protocols.....	2-2
20	2.2.4.1 Commands .....	2-2
21	2.2.4.2 Indications .....	2-2
22	2.3 Protocol Data Unit.....	2-2
23	2.4 Protocol Initialization for the InConfiguration Protocol Instance .....	2-2
24	2.5 Procedures and Messages for the InConfiguration Instance of the Protocol.....	2-3
25	2.5.1 Procedures .....	2-3
26	2.5.2 Commit Procedures .....	2-3
27	2.5.3 Message Formats .....	2-4
28	2.5.3.1 ConfigurationRequest .....	2-4
29	2.5.3.2 ConfigurationResponse.....	2-4
30	2.6 Procedures and Messages for the InUse Instance of the Protocol.....	2-5
31	2.6.1 Procedures .....	2-5
32	2.6.1.1 Access Terminal Requirements .....	2-5

1           2.6.1.1.1 Processing a KeyRequest message.....2-5

2           2.6.1.1.2 Processing the ANKeyComplete message .....2-6

3           2.6.1.2 Access Network Requirements .....2-6

4           2.6.1.3 Session Key Derivation .....2-7

5           2.6.1.4 MICKey, Authentication Key and Encryption Key Generation.....2-7

6       2.6.2 Message Formats.....2-8

7           2.6.2.1 KeyRequest.....2-8

8           2.6.2.2 KeyResponse.....2-9

9           2.6.2.3 ANKeyComplete .....2-12

10          2.6.2.4 AttributeUpdateRequest.....2-13

11          2.6.2.5 AttributeUpdateAccept.....2-14

12          2.6.2.6 AttributeUpdateReject.....2-14

13       2.6.3 Interface to Other Protocols .....2-15

14          2.6.3.1 Commands .....2-15

15          2.6.3.2 Indications.....2-15

16       2.7 Configuration Attributes.....2-15

17       2.8 Protocol Numeric Constants .....2-16

18       2.9 Session State Information.....2-16

19          2.9.1 SKey Parameter .....2-16

20          2.9.2 PMK Parameter.....2-18

21

- 1 No text.

- 1 No figures.

**FIGURES**

- 1 No text.

1 Table 1.6.2.2-1. Definition of Result field..... 1-10  
2 Table 1.6.2.3-1. Definition of Result field..... 1-12  
3 Table 1.7-1. Configurable Values..... 1-14  
4 Table 1.9.1-1. The Format of the Parameter Record for the SKey Parameter ..... 1-15  
5 Table 1.9.2-1. The Format of the Parameter Record for the PMK Parameter..... 1-17  
6 Table 2.6.2.2-1. Definition of Result field.....2-11  
7 Table 2.6.2.3-1. Definition of Result field.....2-13  
8 Table 2.7-1. Configurable Values.....2-15  
9 Table 2.9.1-1. The Format of the Parameter Record for the Skey Parameter .....2-16  
10 Table 2.9.2-1. The Format of the Parameter Record for the PMK Parameter.....2-18  
11

- 1 No text.

**1 FOREWORD**

2 (This foreword is not part of this Standard)

3 This standard was prepared by Technical Specification Group C of the Third Generation  
4 Partnership Project 2 (3GPP2). This standard is evolved from and is a companion to the  
5 cdma2000®<sup>1</sup> standards. This air interface standard defines the Key Exchange Protocols for  
6 the cdma2000 high rate packet data air interface.

---

<sup>1</sup> cdma2000® is the trademark for the technical nomenclature for certain specifications and standards of the Organizational Partners (OPs) of 3GPP2. Geographically (and as of the date of publication), cdma2000® is a registered trademark of the Telecommunications Industry Association (TIA-USA) in the United States.

- 1 The scope of this document covers support for a method for generating and exchanging  
2 Session Key information between the access terminal and the access network. This air  
3 interface standard defines the Key Exchange Protocols for the cdma2000 high rate packet  
4 data air interface
- 5 The Generic Key Exchange Protocol performs the following functions:
- 6 • Proves that both access terminal and access network have the same Pairwise Master  
7 Key.
  - 8 • Derives Session Key(s) from the Pairwise Master Key and nonces that are exchanged  
9 between the access network and access terminal.
  - 10 • Protects against a man-in-the-middle attack where a rogue entity causes the access  
11 terminal and the access network to agree upon a weaker security protocol.

**1 REFERENCES**

2 The following documents contain provisions, which, through reference in this text,  
3 constitute provisions of this document. References are either specific (identified by date of  
4 publication, edition number, version number, etc.) or non-specific. For a specific reference,  
5 subsequent revisions do not apply. For a non-specific reference, the latest version applies.  
6 In the case of a reference to a 3GPP2 document, a non-specific reference implicitly refers to  
7 the latest version of that document in the same Release as the present document.

**8 Normative References**

- 9 [1] 3GPP2 C.S0024-A Version 2.0, cdma2000 High Rate Packet Data Air Interface  
10 Specification.
- 11 [2] 3GPP2 S.S0078-B, Common Security Algorithms.

**12 Informative References**

- 13 [3] 3GPP2 C.R1001, Administration of Parameter Value Assignments for CDMA2000  
14 Spread Spectrum Standards

15

- 1 No text.

## 1 **1 GENERIC KEY EXCHANGE PROTOCOL**

### 2 **1.1 Overview**

3 The Generic Key Exchange Protocol belongs to the Security Layer of the cdma2000 High  
4 Rate Packet Data air interface defined in [1]. The Generic Key Exchange Protocol provides a  
5 method for generating and exchanging Session Key between the access terminal and the  
6 access network based on a Pairwise Master Key. The Pairwise Master Key is negotiated by  
7 higher layer protocols.

8 The Generic Key Exchange Protocol performs the following functions:

- 9 • Proves that both access terminal and access network have the same Pairwise  
10 Master Key.
- 11 • Derives Session Key(s) from the Pairwise Master Key and nonces that are exchanged  
12 between the access network and access terminal.
- 13 • Protects against a man-in-the-middle attack where a rogue entity causes the access  
14 terminal and the access network to agree upon a weaker security protocol.

15 The Generic Key Exchange Protocol can generate multiple Session Keys from the Pairwise  
16 Master Key and store the Session Keys as part of the session. This allows the access  
17 network and the access terminal to later switch to a new Session Key without having to  
18 execute the key exchange procedures again.

### 19 **1.2 Primitives and Public Data**

#### 20 1.2.1 Commands

21 This protocol does not define any commands.

#### 22 1.2.2 Return Indications

23 This protocol does not return any indications.

#### 24 1.2.3 Public Data

- 25 • Subtype for this protocol
- 26 • FACAuthKey and its length  
27 The authentication key for use on Forward Assigned Channels (e.g., the Forward Traffic  
28 Channel).
- 29 • RACAuthKey and its length  
30 The authentication key for use on Reverse Assigned Channels (e.g., the Reverse Traffic  
31 Channel).
- 32 • FACEncKey and its length  
33 The encryption key for use on Forward Assigned Channels (e.g., the Forward Traffic  
34 Channel).

- 1 • RACEncKey and its length  
2 The encryption key for use on Reverse Assigned Channels (e.g., the Reverse Traffic  
3 Channel).
- 4 • FPCAuthKey and its length  
5 The authentication key for use on Forward Public Channels (e.g., the Control Channel).
- 6 • RPCAuthKey and its length  
7 The authentication key for use on Reverse Public Channels (e.g., the Access Channel).
- 8 • FPCEncKey and its length  
9 The encryption key for use on Forward Public Channels (e.g. the Control Channel).
- 10 • RPCEncKey and its length  
11 The encryption key for use on Reverse Public Channels (e.g. the Access Channel).

## 12 1.2.4 Interface to Other Protocols

### 13 1.2.4.1 Commands

14 This protocol does not define any commands.

### 15 1.2.4.2 Indications

16 This protocol does not register to receive any indications.

## 17 **1.3 Protocol Data Unit**

18 The transmission unit of this protocol is a message. This is a control protocol and,  
19 therefore, it does not carry payload on behalf of other layers or protocols.

20 This protocol uses the Signaling Application to transmit and receive messages.

## 21 **1.4 Protocol Initialization for the InConfiguration Protocol Instance**

22 Upon creation, the InConfiguration instance of this protocol in the access terminal and the  
23 access network shall perform the following in the order specified:

- 24 • The fall-back values of the attributes for this protocol instance shall be set to the  
25 default values specified for each attribute.
- 26 • If the InUse instance of this protocol has the same protocol subtype as this  
27 InConfiguration protocol instance, then
  - 28 – The access terminal and the access network shall set the fall-back values of  
29 the attributes defined by the InConfiguration protocol instance to the values of  
30 the corresponding attributes associated with the InUse protocol instance.
  - 31 – The access terminal and the access network shall set the value of the public  
32 data associated with the InConfiguration protocol instance to the  
33 corresponding public data values for the InUse protocol.
- 34 • The value for each attribute for this protocol instance shall be set to the fall-back value  
35 for that attribute.

## 1 1.5 Procedures and Messages for the InConfiguration Instance of the Protocol

### 2 1.5.1 Procedures

3 This protocol uses the Generic Configuration Protocol (see [1]) to define the processing of  
4 the configuration messages.

### 5 1.5.2 Commit Procedures

6 The access terminal and the access network shall perform the procedures specified in this  
7 section, in the order specified, when directed by the InUse instance of the Session  
8 Configuration Protocol to execute the Commit procedures:

- 9 • All the public data that are defined by this protocol, but are not defined by the InUse  
10 protocol instance shall be added to the public data of the InUse protocol.
- 11 • If the InUse instance of this protocol has the same subtype as this protocol instance,  
12 then
  - 13 – The access terminal and the access network shall set the attribute values  
14 associated with the InUse instance of this protocol to the attribute values  
15 associated with the InConfiguration instance of this protocol,
  - 16 – The access terminal and the access network shall purge the InConfiguration  
17 instance of the protocol.
- 18 • If the InUse instance of this protocol does not have the same subtype as this protocol  
19 instance, then the access terminal and the access network shall perform the following:
  - 20 – Set SKey[*i*] to zero and its length to  $N_{\text{GKEPSessionKeyLen}}$ , for values of *i* from 0  
21 through 7.
  - 22 – Set SKeyTemp[*i*] to zero and its length to  $N_{\text{GKEPSessionKeyLen}}$ , for values of *i* from 0  
23 through 7.
  - 24 – Set FACAAuthKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
25 7.
  - 26 – Set RACAAuthKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
27 7.
  - 28 – Set FACEncKey[*i*] to zero and its length to 128, for values of *i* from 0 through 7.
  - 29 – Set RACEncKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
30 7.
  - 31 – Set FPCAAuthKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
32 7.
  - 33 – Set RPCAAuthKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
34 7.
  - 35 – Set FPCEncKey[*i*] to zero and its length to 128, for values of *i* from 0 through 7.
  - 36 – Set RPCEncKey[*i*] to zero and its length to 128, for values of *i* from 0 through 7.

- 1           – Set ATNonce to a 128-bit random number generated according to the  
2 pseudorandom number generator specified in [1].
- 3           – Set ANNonce to a 128-bit random number.
- 4           – The InConfiguration protocol instance shall become the InUse protocol  
5 instance for the Key Exchange Protocol.
- 6 • All the public data not defined by this protocol shall be removed from the public data of  
7 the InUse protocol.

### 8 1.5.3 Message Formats

#### 9 1.5.3.1 ConfigurationRequest

10 The ConfigurationRequest message format is as follows:

11

Field	Length (bits)
MessageID	8
TransactionID	8
Zero or more instances of the following record	
AttributeRecord	Attribute dependent

- 12 MessageID           The sender shall set this field to 0x50.
- 13 TransactionID       The sender shall increment this value for each new  
14 ConfigurationRequest message sent.
- 15 AttributeRecord     The format of this record is specified in [1].

16

<b>Channels</b>	FTC    RTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

#### 17 1.5.3.2 ConfigurationResponse

18 The ConfigurationResponse message format is as follows:

19

Field	Length (bits)
MessageID	8
TransactionID	8

Zero or more instances of the following record

AttributeRecord	Attribute dependent
-----------------	---------------------

- 1 MessageID            The sender shall set this field to 0x51.
- 2 TransactionID        The sender shall set this value to the TransactionID field of the
- 3                            corresponding ConfigurationRequest message.
- 4 AttributeRecord      An attribute record containing a single attribute value. If this
- 5                            message selects a complex attribute, only the ValueID field of the
- 6                            complex attribute shall be included in the message. The format of the
- 7                            AttributeRecord is given in [1]. The sender shall not include more
- 8                            than one attribute record with the same attribute identifier.
- 9

<b>Channels</b>	FTC    RTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

10 **1.6 Procedures and Messages for the InUse Instance of the Protocol**

11 1.6.1 Procedures

12 The Generic Key Exchange Protocol derives secret session keys, verifies that the access  
 13 terminal and the access network have derived the same session keys, and exchanges  
 14 security capabilities. The key exchange procedure uses the KeyRequest, KeyResponse, and  
 15 ANKeyComplete messages.

16 1.6.1.1 Access Terminal Requirements

17 1.6.1.1.1 Processing a KeyRequest message

18 Upon receiving the KeyRequest message, the access terminal shall perform the following in  
 19 the order in which the requirements are specified:

- 20 • The access terminal shall identify the PairwiseMasterKey that satisfies
- 21 PairwiseMasterKeyID = ehmacsha(key=PairwiseMasterKey, key\_length=length of
- 22 PairwiseMasterKey in units of octets, message= "PairwiseMasterKeyID",
- 23 message\_length=length of message in units of bits, message\_offset=0, MAC\_length=16),
- 24 where PairwiseMasterKeyID is a field of the received KeyRequest message,
- 25 "PairwiseMasterKeyID" is the ASCII encoded value of the string, and the ehmacsha
- 26 function is specified in [2].

- 1 • If the access terminal cannot identify a valid PairwiseMasterKey that satisfies the above
- 2 Equation, then the access terminal shall send a KeyResponse message with Result set
- 3 to 0x03, declare failure, and stop performing the rest of the key exchange procedure.
- 4 • The access terminal shall set ATNonce to  $(\text{ATNonce} + 1) \bmod 2^{128}$ .
- 5 • The access terminal shall compute SKeyTemp[*i*] the session key as specified in 1.6.1.3.
- 6 • The access terminal shall generate MICKey[*i*] from SKeyTemp[*i*] as specified in 1.6.1.4.
- 7 • The access terminal shall send a KeyResponse message.

#### 8 1.6.1.1.2 Processing the ANKeyComplete message

9 After receiving an ANKeyComplete message with a TransactionID field that matches the  
10 TransactionID field of the associated KeyRequest message, the access terminal shall  
11 perform the following in the order in which the requirements are specified:

- 12 • The access terminal shall generate a MessageIntegrityCode as  $\text{ehmacsha}(\text{key}=\text{MICKey}[i],$   
13  $\text{key\_length}=\text{length of MICKey}[i], \text{message}=\text{Message}, \text{message\_length}=\text{length of Message}$   
14  $\text{in units of bits}, \text{message\_offset}=0, \text{MAC\_length}=10)$ , where *Message* is the received  
15 ANKeyComplete message with the MessageIntegrityCode field set to zero, *i* is the  
16 SessionKeyIndex field of the corresponding KeyRequest message, and the ehmacsha  
17 function is specified in [2].
- 18 • If the MessageIntegrityCode computed in the previous step does not match the  
19 MessageIntegrityCode field of ANKeyComplete message, then the access terminal shall  
20 declare failure, discard the ANKeyComplete message, and stop performing the rest of  
21 the key exchange procedure.
- 22 • If the Result field of the ANKeyComplete message is not 0x00, the access terminal shall  
23 declare failure, and stop performing the rest of the key exchange procedures.
- 24 • The access terminal shall set SKey[*i*] to SKeyTemp[*i*], and shall generate Authentication  
25 Key and Encryption Key as defined in 1.6.1.4.

#### 26 1.6.1.2 Access Network Requirements

27 The access network shall initiate the key exchange procedure by sending a KeyRequest  
28 message. The access network shall set ANNonce to a 128-bit random number.

29 After receiving a KeyResponse message with a TransactionID field that matches the  
30 TransactionID field of the associated KeyRequest message, the access network shall  
31 perform the following:

- 32 • If the Result field of the KeyResponse message is not 0x00, then the access network  
33 shall declare failure and stop performing the rest of the key exchange procedure.
- 34 • The access network shall compute SKeyTemp[*i*], the session key as specified in 1.6.1.3.
- 35 • The access network shall generate MICKey as specified in 1.6.1.4.

- 1 • The access network shall generate a MessageIntegrityCode as ehmacsha  
2 (key=MICKey[*i*], key\_length=16, message=*Message*, message\_length=length of *Message*  
3 in units of bits, message\_offset=0, MAC\_length=10), where *Message* is the received  
4 KeyResponse message with the MessageIntegrityCode field set to zero, *i* is the  
5 SessionKeyIndex field of the corresponding KeyRequest message, and the ehmacsha  
6 function is specified in [2].
- 7 • If the MessageIntegrityCode computed in the previous step does not match the  
8 MessageIntegrityCode field of KeyResponse message, or if the supported protocol  
9 subtypes sent by the access terminal in the KeyResponse message contain a protocol  
10 (or set of protocols) that the access network supports and prefers to use over the  
11 subtypes that have been negotiated, then the access network shall declare failure, send  
12 an ANKeyComplete message with the appropriate Result, and stop performing the rest  
13 of the key exchange procedure. Otherwise, the access network shall send an  
14 ANKeyComplete message.
- 15 • The access network shall set SKey[*i*] to SKeyTemp[*i*] and generate Authentication Key  
16 and Encryption Key as specified in 1.6.1.4.

### 17 1.6.1.3 Session Key Derivation

18 The access terminal and the access network shall derive SKey[*i*] as follows:

- 19 • Set *j* to an 8-bit number with value zero.
- 20 • while  $j < N_{\text{GKEPSessionKeyLen}}/128$
- 21     – Set SKeyTemp[*i*] to  $N_{\text{GKEPSessionKeyLen}}$  least significant bits of { SKeyTemp[*i*] | 128  
22 most significant bits of ehmacsha(key=PairwiseMasterKey, key\_length=length  
23 of PairwiseMasterKey in units of octets, message=ATNonce|ANNonce|*j*,  
24 message\_length= length of message in units of bits, message\_offset=0,  
25 MAC\_length=16) }, where the ehmacsha function is specified in [2], and *j* is  
26 represented as an 8-bit field.
- 27     – Set *j* to *j*+1.

### 28 1.6.1.4 MICKey, Authentication Key and Encryption Key Generation

29 The keys used for MessageIntegrityCode, authentication and encryption are generated from  
30 the session key using the procedures specified in this section.

31 The access network and the access terminal shall compute and store a MICKey,  
32 Authentication Key, and Encryption Key, derived from each session key. The keys derived  
33 from SKey[*i*] or SKeyTemp[*i*] are referred to by the subscript *i*. The access network and the  
34 access terminal shall use the Authentication Key and Encryption Key derived from the  
35 SKey with index *i*, where *i* is the value of the InUseSessionKeyIndex attribute.

36 The access network and the access terminal shall set the MICKey[*i*] to SKeyTemp[*i*][127:0],  
37 where *i* is the session key index.

38 The access network and the access terminal shall set FACAuthKey[*i*], FPCAuthKey[*i*],  
39 RACAuthKey[*i*], and RPCAuthKey[*i*] to SKey[*i*][255:128], where *i* is the session key index.

1 The access network and the access terminal shall set FACEncKey[*i*], FPCEncKey[*i*,  
2 RACEncKey[*i*], and RPCEncKey[*i*] to SKey[*i*][383:256], where *i* is the session key index.

### 3 1.6.2 Message Formats

#### 4 1.6.2.1 KeyRequest

5 The access network sends the KeyRequest message to initiate the session key exchange.  
6

Field	Length (bits)
MessageID	8
TransactionID	8
SessionKeyIndex	3
Reserved	5
PairwiseMasterKeyID	128
ANNonce	128

7	MessageID	The access network shall set this field to 0x00.
8	TransactionID	The access network shall set this field to a unique value not being
9		used as TransactionID in any other on-going Key Exchange with the
10		access terminal.
11	SessionKeyIndex	The access network shall set this field to the ID of the SKey for which
12		this key exchange is being initiated.
13	Reserved	The access network shall set this field to '00000'. The access terminal
14		shall ignore this field.
15	PairwiseMasterKeyID	
16		The access network shall set this field to
17		ehmacsha(key=PairwiseMasterKey, key_length=length of
18		PairwiseMasterKey in units of octets, message=
19		"PairwiseMasterKeyID", message_length=length of message in units
20		of bits, message_offset=0, MAC_length=16), where
21		PairwiseMasterKeyID is a field of the received KeyRequest message,
22		"PairwiseMasterKeyID" is the ASCII encoded value of the string, and
23		the ehmacsha function is specified in [2].
24	ANNonce	The access network shall set this field to the nonce chosen by the
25		access network.
26		

<b>Channels</b>	FTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

## 1 1.6.2.2 KeyResponse

2 The access terminal sends the KeyResponse message in response to the KeyRequest  
3 message.

4

<b>Field</b>	<b>Length (bits)</b>
MessageID	8
TransactionID	8
Result	8
ATNonce	0 or 128
SecuritySubtypeCount	0 or 8

Zero or SecuritySubtypeCount occurrences of the following field:

SecuritySubtype	16
-----------------	----

AuthenticationSubtypeCount	0 or 8
AuthenticationSubtypeCount occurrences of the following field:	
AuthenticationSubtype	16

EncryptionSubtypeCount	0 or 8
EncryptionSubtypeCount occurrences of the following field:	
EncryptionSubtype	16

KeyExchangeSubtypeCount	0 or 8
KeyExchangeSubtypeCount occurrences of the following field:	
KeyExchangeSubtype	16

MessageIntegrityCode	0 or 80
----------------------	---------

5 MessageID The access terminal shall set this field to 0x01.

6 TransactionID The access terminal shall set this field to the value of the  
7 TransactionID field of the KeyRequest message to which the access  
8 terminal is responding.

1 Result The access terminal shall set this field according to Table 1.6.2.2-1.

2 **Table 1.6.2.2-1. Definition of Result field**

Value	Meaning
0x00	Continue key exchange.
0x03	PairwiseMasterKey not found.
All other values	Reserved

3 ATNonce If Result is 0x00, then the access terminal shall set this field to the  
4 nonce chosen by the access terminal. Otherwise, the access terminal  
5 shall omit this field.

6 SecuritySubtypeCount

7 If Result is 0x00, then the access terminal shall set this field to the  
8 number of subtypes of the Security Protocol supported by the access  
9 terminal besides the HardLink subtype and besides the default  
10 subtype. Otherwise, the access terminal shall omit this field.

11 SecuritySubtype

12 If Result is 0x00, then the access terminal shall set this field to the  
13 subtype of the Security Protocol supported by the access terminal.  
14 Otherwise, the access terminal shall omit this field. The access  
15 terminal shall not set this field to 0xfffe (HardLink subtype). The  
access terminal shall not set this field to 0x0000.

16 AuthenticationSubtypeCount

17 If Result is 0x00, then the access terminal shall set this field to the  
18 number of subtypes of the Authentication Protocol supported by the  
19 access terminal besides the HardLink subtype and besides the  
20 default subtype. Otherwise, the access terminal shall omit this field.

21 AuthenticationSubtype

22 If Result is 0x00, then the access terminal shall set this field to the  
23 subtype of the Authentication Protocol supported by the access  
24 terminal. Otherwise, the access terminal shall omit this field. The  
25 access terminal shall not set this field to 0xfffe (HardLink subtype).  
26 The access terminal shall not set this field to 0x0000.

27 EncryptionSubtypeCount

28 If Result is 0x00, then the access terminal shall set this field to the  
29 number of subtypes of the Encryption Protocol supported by the  
30 access terminal besides the HardLink subtype and besides the  
31 default subtype. Otherwise, the access terminal shall omit this field.

- 1 EncryptionSubtype If Result is 0x00, then the access terminal shall set this field to the  
 2 subtype of the Encryption Protocol supported by the access terminal.  
 3 Otherwise, the access terminal shall omit this field. The access  
 4 terminal shall not set this field to 0xfffe (HardLink subtype). The  
 5 access terminal shall not set this field to 0x0000.
- 6 KeyExchangeSubtypeCount  
 7 If Result is 0x00, then the access terminal shall set this field to the  
 8 number of subtypes of the Key Exchange Protocol supported by the  
 9 access terminal besides the HardLink subtype and besides the  
 10 default subtype. Otherwise, the access terminal shall omit this field.
- 11 KeyExchangeSubtype  
 12 If Result is 0x00, then the access terminal shall set this field to the  
 13 subtype of the Key Exchange Protocol supported by the access  
 14 terminal. Otherwise, the access terminal shall omit this field. The  
 15 access terminal shall not set this field to 0xfffe (HardLink subtype).  
 16 The access terminal shall not set this field to 0x0000.
- 17 MessageIntegrityCode  
 18 If Result is 0x00, then the access terminal shall set this field to  
 19 ehmacsha(key=MICKey[*i*], key\_length= length of MICKey[*i*] in units of  
 20 octets, message=*Message*, message\_length= length of *Message* in  
 21 units of bits, message\_offset=0, MAC\_length=10), where *Message* is  
 22 set to all fields of this message with this field set to zero, *i* is  
 23 SessionKeyIndex field of the corresponding KeyRequest message, and  
 24 the ehmacsha function is specified in [2]. Otherwise, the access  
 25 terminal shall omit this field.  
 26

<b>Channels</b>	RTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

### 27 1.6.2.3 ANKeyComplete

- 28 The access network sends the ANKeyComplete message in response to the KeyResponse  
 29 message.  
 30

Field	Length (bits)
MessageID	8
TransactionID	8
Result	8
MessageIntegrityCode	80

- 31 MessageID The access network shall set this field to 0x02.

1 TransactionID The access network shall set this field to the value of the  
2 TransactionID field of the corresponding KeyResponse message.

3 Result The access network shall set this field according to Table 1.6.2.3-1.

4 **Table 1.6.2.3-1. Definition of Result field**

Value	Meaning
0x00	Key exchange successful.
0x01	MessageIntegrityCode failed.
0x02	MessageIntegrityCode successful, but security capabilities verification failed.
All other values	Reserved

5 MessageIntegrityCode  
6 The access network shall set this field to ehmacsha(key=MICKey[i],  
7 key\_length= length of MICKey[i] in units of octets, message=*Message*,  
8 message\_length= length of *Message* in units of bits,  
9 message\_offset=0, MAC\_length=10), where *Message* is set to all fields  
10 of this message with this field set to zero, *i* is the SessionKeyIndex  
11 field of the corresponding KeyRequest message, and the ehmacsha  
12 function is specified in [2].  
13

<b>Channels</b>	FTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

#### 14 1.6.2.4 AttributeUpdateRequest

15 The sender sends an AttributeUpdateRequest message to offer a set of attribute values for a  
16 given attribute.  
17

Field	Length (bits)
MessageID	8
TransactionID	8

One or more instances of the following record

AttributeRecord	Attribute dependent
-----------------	---------------------

18 MessageID The sender shall set this field to 0x52.

19 TransactionID The sender shall increment this value for each new  
20 AttributeUpdateRequest message sent.

21 AttributeRecord The format of this record is specified in [1].

1

<b>Channels</b>	CC	FTC	RTC	<b>SLP</b>	Reliable on FTC and RTC Best Effort on CC
<b>Addressing</b>	unicast			<b>Priority</b>	40

## 2 1.6.2.5 AttributeUpdateAccept

3 The sender sends an AttributeUpdateAccept message in response to an  
4 AttributeUpdateRequest message to select an attribute value from a list of offered values.

5

Field	Length (bits)
MessageID	8
TransactionID	8

6 MessageID The sender shall set this field to 0x53.

7 TransactionID The sender shall set this value to the TransactionID field of the  
8 corresponding AttributeUpdateRequest message.

9

<b>Channels</b>	AC	FTC	RTC	<b>SLP</b>	Reliable on FTC and RTC Best Effort on AC
<b>Addressing</b>	unicast			<b>Priority</b>	40

## 10 1.6.2.6 AttributeUpdateReject

11 The access network sends an AttributeUpdateReject message to reject the set of attribute  
12 values proposed by the access terminal in the corresponding AttributeUpdateRequest  
13 message.

14

Field	Length (bits)
MessageID	8
TransactionID	8

15 MessageID The access network shall set this field to 0x54.

16 TransactionID The access network shall set this value to the TransactionID field of  
17 the corresponding AttributeUpdateRequest message.

18

<b>Channels</b>	FTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

## 1 1.6.3 Interface to Other Protocols

## 2 1.6.3.1 Commands

3 This protocol does not issue any commands.

## 4 1.6.3.2 Indications

5 This protocol does not register to receive any indications.

6 **1.7 Configuration Attributes**

7 The configurable simple attribute for this protocol is listed in Table 1.7-1.

8 The access terminal shall use as defaults the values in Table 1.7-1 that are typed in **bold**  
9 **italics**.

10 Unless specified otherwise, the access terminal and the access network shall not use the  
11 Generic Attribute Update Protocol to update configurable attributes belonging to the  
12 Generic Key Exchange Protocol. The access terminal and the access network shall support  
13 the use of the Generic Attribute Update Protocol to update values of the  
14 InUseSessionKeyIndex attribute. The access terminal shall not send an  
15 AttributeUpdateRequest message containing the InUseSessionKeyIndex attribute. The  
16 access network may include the InUseSessionKeyIndex attribute in an  
17 AttributeUpdateRequest message sent on the Control Channel.

18 **Table 1.7-1. Configurable Values**

<b>Attribute ID</b>	<b>Attribute</b>	<b>Values</b>	<b>Meaning</b>
0x00	InUseSessionKeyIndex	<b>0x00</b>	SKey[0] is used.
		0x01	SKey[1] is used
		0x02	SKey[2] is used
		0x03	SKey[3] is used
		0x04	SKey[4] is used
		0x05	SKey[5] is used
		0x06	SKey[6] is used
		0x07	SKey[7] is used
		0x08-0xff	Reserved

1 **1.8 Protocol Numeric Constants**

Constant	Meaning	Value
N <sub>KEPType</sub>	Type field for this protocol	See [3]
N <sub>GKEP</sub>	Subtype field for this protocol	0x0002
N <sub>GKEPSessionKeyLen</sub>	Length of Session Key in units of bits	384

2 **1.9 Session State Information**

3 The Session State Information record (see [1]) consists of parameter records.

4 This protocol defines the following parameter record in addition to the configuration  
5 attributes for this protocol.

## 6 1.9.1 SKey Parameter

7 **Table 1.9.1-1. The Format of the Parameter Record for the SKey Parameter**

Field	Length (bits)
ParameterType	8
Length	8
SKey[0]Included	1
SKey[1]Included	1
SKey[2]Included	1
SKey[3]Included	1
SKey[4]Included	1
SKey[5]Included	1
SKey[6]Included	1
SKey[7]Included	1
SKey[0]	0 or 128
SKey[1]	0 or 128
SKey[2]	0 or 128
SKey[3]	0 or 128
SKey[4]	0 or 128
SKey[5]	0 or 128
SKey[6]	0 or 128
SKey[7]	0 or 128

8 ParameterType This field shall be set to 0x01 for this parameter record.

9 Length This field shall be set to the length of this parameter record in units  
10 of octets excluding the Length field.

1	SKey[0]Included	If SKey[0] is zero, then this field shall be set to '0'. Otherwise, this
2		field shall be set to '1'.
3	SKey[1]Included	If SKey[1] is zero, then this field shall be set to '0'. Otherwise, this
4		field shall be set to '1'.
5	SKey[2]Included	If SKey[2] is zero, then this field shall be set to '0'. Otherwise, this
6		field shall be set to '1'.
7	SKey[3]Included	If SKey[3] is zero, then this field shall be set to '0'. Otherwise, this
8		field shall be set to '1'.
9	SKey[4]Included	If SKey[4] is zero, then this field shall be set to '0'. Otherwise, this
10		field shall be set to '1'.
11	SKey[5]Included	If SKey[5] is zero, then this field shall be set to '0'. Otherwise, this
12		field shall be set to '1'.
13	SKey[6]Included	If SKey[6] is zero, then this field shall be set to '0'. Otherwise, this
14		field shall be set to '1'.
15	SKey[7]Included	If SKey[7] is zero, then this field shall be set to '0'. Otherwise, this
16		field shall be set to '1'.
17	SKey[0]	If SKey[0]Included is '0', then this field shall be omitted. Otherwise,
18		this field shall be set to the value of the session key with key index
19		0x00.
20	SKey[1]	If SKey[1]Included is '0', then this field shall be omitted. Otherwise,
21		this field shall be set to the value of the session key with key index
22		0x01.
23	SKey[2]	If SKey[2]Included is '0', then this field shall be omitted. Otherwise,
24		this field shall be set to the value of the session key with key index
25		0x02.
26	SKey[3]	If SKey[3]Included is '0', then this field shall be omitted. Otherwise,
27		this field shall be set to the value of the session key with key index
28		0x03.
29	SKey[4]	If SKey[4]Included is '0', then this field shall be omitted. Otherwise,
30		this field shall be set to the value of the session key with key index
31		0x04.
32	SKey[5]	If SKey[5]Included is '0', then this field shall be omitted. Otherwise,
33		this field shall be set to the value of the session key with key index
34		0x05.

- 1 SKey[6] If SKey[6]Included is '0', then this field shall be omitted. Otherwise,  
 2 this field shall be set to the value of the session key with key index  
 3 0x06.
- 4 SKey[7] If SKey[7]Included is '0', then this field shall be omitted. Otherwise,  
 5 this field shall be set to the value of the session key with key index  
 6 0x07.

7 1.9.2 PMK Parameter

8 **Table 1.9.2-1. The Format of the Parameter Record for the PMK Parameter**

Field	Length (bits)
ParameterType	8
Length	8
PMKCount	8
PMKCount occurrences of the following two fields:	
PMKLength	8
PMK	PMKLength × 8

- 9 ParameterType This field shall be set to 0x04 for this parameter record.
- 10 Length This field shall be set to the length of this parameter record in units  
 11 of octets excluding the Length field.
- 12 PMKCount This field shall be set to the number of occurrences of the PMK field  
 13 in this parameter record.
- 14 PMKLength This field shall be set to the length of the PMK field in units of octets.
- 15 PMK This field shall be set to a PairwiseMasterKey.

- 1 No text.

## 1 **2 MUTLI-KEY KEY EXCHANGE PROTOCOL**

### 2 **2.1 Overview**

3 The Multi-Key Key Exchange Protocol belongs to the Security Layer of the cdma2000 High  
4 Rate Packet Data air interface defined in [1]. The Multi-Key Key Exchange Protocol provides  
5 a method for generating and exchanging Session Keys between the access terminal and the  
6 access network based on a Pairwise Master Key. The Pairwise Master Key is negotiated by  
7 higher layer protocols.

8 The Multi-Key Key Exchange Protocol performs the following functions:

- 9 • Proves that both access terminal and access network have the same Pairwise  
10 Master Key.
- 11 • Derives Session Key(s) from the Pairwise Master Key and nonces that are exchanged  
12 between the access network and access terminal.
- 13 • Protects against a man-in-the-middle attack where a rogue entity causes the access  
14 terminal and the access network to agree upon a weaker security protocol.

15 The Multi-Key Key Exchange Protocol can generate multiple Session Keys from the Pairwise  
16 Master Key and store the Session Keys as part of the session. This allows the access  
17 network and the access terminal to later switch to a new Session Key without having to  
18 execute the key exchange procedures multiple times.

### 19 **2.2 Primitives and Public Data**

#### 20 2.2.1 Commands

21 This protocol does not define any commands.

#### 22 2.2.2 Return Indications

23 This protocol does not return any indications.

#### 24 2.2.3 Public Data

- 25 • Subtype for this protocol
- 26 • FACAuthKey and its length  
27 The authentication key for use on Forward Assigned Channels (e.g., the Forward Traffic  
28 Channel).
- 29 • RACAuthKey and its length  
30 The authentication key for use on Reverse Assigned Channels (e.g., the Reverse Traffic  
31 Channel).
- 32 • FACEncKey and its length  
33 The encryption key for use on Forward Assigned Channels (e.g., the Forward Traffic  
34 Channel).

- 1 • RACEncKey and its length  
2 The encryption key for use on Reverse Assigned Channels (e.g., the Reverse Traffic  
3 Channel).
- 4 • FPCAuthKey and its length  
5 The authentication key for use on Forward Public Channels (e.g., the Control Channel).
- 6 • RPCAuthKey and its length  
7 The authentication key for use on Reverse Public Channels (e.g., the Access Channel).
- 8 • FPCEncKey and its length  
9 The encryption key for use on Forward Public Channels (e.g. the Control Channel).
- 10 • RPCEncKey and its length  
11 The encryption key for use on Reverse Public Channels (e.g. the Access Channel).

## 12 2.2.4 Interface to Other Protocols

### 13 2.2.4.1 Commands

14 This protocol does not define any commands.

### 15 2.2.4.2 Indications

16 This protocol does not register to receive any indications.

## 17 **2.3 Protocol Data Unit**

18 The transmission unit of this protocol is a message. This is a control protocol and,  
19 therefore, it does not carry payload on behalf of other layers or protocols.

20 This protocol uses the Signaling Application to transmit and receive messages.

## 21 **2.4 Protocol Initialization for the InConfiguration Protocol Instance**

22 Upon creation, the InConfiguration instance of this protocol in the access terminal and the  
23 access network shall perform the following in the order specified:

- 24 • The fall-back values of the attributes for this protocol instance shall be set to the  
25 default values specified for each attribute.
- 26 • If the InUse instance of this protocol has the same protocol subtype as this  
27 InConfiguration protocol instance, then
  - 28 – The access terminal and the access network shall set the fall-back values of  
29 the attributes defined by the InConfiguration protocol instance to the values of  
30 the corresponding attributes associated with the InUse protocol instance.
  - 31 – The access terminal and the access network shall set the value of the public  
32 data associated with the InConfiguration protocol instance to the  
33 corresponding public data values for the InUse protocol.
- 34 • The value for each attribute for this protocol instance shall be set to the fall-back value  
35 for that attribute.

## 1 **2.5 Procedures and Messages for the InConfiguration Instance of the Protocol**

### 2 2.5.1 Procedures

3 This protocol uses the Generic Configuration Protocol (see [1]) to define the processing of  
4 the configuration messages.

### 5 2.5.2 Commit Procedures

6 The access terminal and the access network shall perform the procedures specified in this  
7 section, in the order specified, when directed by the InUse instance of the Session  
8 Configuration Protocol to execute the Commit procedures:

- 9 • All the public data that are defined by this protocol, but are not defined by the InUse  
10 protocol instance shall be added to the public data of the InUse protocol.
- 11 • If the InUse instance of this protocol has the same subtype as this protocol instance,  
12 then
  - 13 – The access terminal and the access network shall set the attribute values  
14 associated with the InUse instance of this protocol to the attribute values  
15 associated with the InConfiguration instance of this protocol,
  - 16 – The access terminal and the access network shall purge the InConfiguration  
17 instance of the protocol.
- 18 • If the InUse instance of this protocol does not have the same subtype as this protocol  
19 instance, then the access terminal and the access network shall perform the following:
  - 20 – Set SKey[*i*] to zero and its length to  $N_{\text{MKEPSessionKeyLen}}$ , for values of *i* from 0  
21 through 7.
  - 22 – Set SKeyTemp[*i*] to zero and its length to  $N_{\text{MKEPSessionKeyLen}}$ , for values of *i* from 0  
23 through 7.
  - 24 – Set FACAuthKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
25 7.
  - 26 – Set RACAuthKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
27 7.
  - 28 – Set FACEncKey[*i*] to zero and its length to 128, for values of *i* from 0 through 7.
  - 29 – Set RACEncKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
30 7.
  - 31 – Set FPCAuthKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
32 7.
  - 33 – Set RPCAuthKey[*i*] to zero and its length to 128, for values of *i* from 0 through  
34 7.
  - 35 – Set FPCEncKey[*i*] to zero and its length to 128, for values of *i* from 0 through 7.
  - 36 – Set RPCEncKey[*i*] to zero and its length to 128, for values of *i* from 0 through 7.

- 1           – Set ATNonce to a 128-bit random number generated according to the  
2 pseudorandom number generator specified in [1].
- 3           – Set ANNonce to a 128-bit random number.
- 4           – The InConfiguration protocol instance shall become the InUse protocol  
5 instance for the Key Exchange Protocol.
- 6 • All the public data not defined by this protocol shall be removed from the public data of  
7 the InUse protocol.

### 8 2.5.3 Message Formats

#### 9 2.5.3.1 ConfigurationRequest

10 The ConfigurationRequest message format is as follows:

11

Field	Length (bits)
MessageID	8
TransactionID	8
Zero or more instances of the following record	
AttributeRecord	Attribute dependent

12 MessageID           The sender shall set this field to 0x50.

13 TransactionID       The sender shall increment this value for each new  
14 ConfigurationRequest message sent.

15 AttributeRecord     The format of this record is specified in [1].

16

<b>Channels</b>	FTC    RTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

#### 17 2.5.3.2 ConfigurationResponse

18 The ConfigurationResponse message format is as follows:

19

Field	Length (bits)
MessageID	8
TransactionID	8

Zero or more instances of the following record

AttributeRecord	Attribute dependent
-----------------	---------------------

- 1 MessageID            The sender shall set this field to 0x51.
- 2 TransactionID        The sender shall set this value to the TransactionID field of the
- 3                            corresponding ConfigurationRequest message.
- 4 AttributeRecord      An attribute record containing a single attribute value. If this
- 5                            message selects a complex attribute, only the ValueID field of the
- 6                            complex attribute shall be included in the message. The format of the
- 7                            AttributeRecord is given in [1]. The sender shall not include more
- 8                            than one attribute record with the same attribute identifier.
- 9

<b>Channels</b>	FTC    RTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

10 **2.6 Procedures and Messages for the InUse Instance of the Protocol**

11 2.6.1 Procedures

12 The Multi-Key Key Exchange Protocol derives secret session keys, verifies that the access  
 13 terminal and the access network have derived the same session keys, and exchanges  
 14 security capabilities. The key exchange procedure uses the KeyRequest, KeyResponse, and  
 15 ANKeyComplete messages.

16 2.6.1.1 Access Terminal Requirements

17 2.6.1.1.1 Processing a KeyRequest message

18 Upon receiving the KeyRequest message, the access terminal shall perform the following in  
 19 the order in which the requirements are specified:

- 20 • The access terminal shall identify the PairwiseMasterKey that satisfies
- 21 PairwiseMasterKeyID = 128 most significant bits of
- 22 ehmacsha256(key=PairwiseMasterKey, key\_length=length of PairwiseMasterKey in units
- 23 of octets, message= "PairwiseMasterKeyID", message\_length=length of message in units
- 24 of bits, message\_offset=0, MAC\_length=16), where PairwiseMasterKeyID is a field of the
- 25 received KeyRequest message, "PairwiseMasterKeyID" is the ASCII encoded value of the
- 26 string, and the ehmacsha256 function is specified in [2].

- 1 • If the access terminal cannot identify a valid PairwiseMasterKey that satisfies the above
- 2 Equation, then the access terminal shall send a KeyResponse message with Result set
- 3 to 0x03, declare failure, and stop performing the rest of the key exchange procedure.
- 4 • The access terminal shall set ATNonce to  $(\text{ATNonce} + 1) \bmod 2^{128}$ .
- 5 • The access terminal shall compute SKeyTemp[*i*] the temporary session key as specified
- 6 in 2.6.1.3.
- 7 • The access terminal shall generate MICKey[*i*] from SKeyTemp[*i*] as specified in 2.6.1.4.
- 8 • The access terminal shall send a KeyResponse message.

#### 9 2.6.1.1.2 Processing the ANKeyComplete message

10 After receiving an ANKeyComplete message with a TransactionID field that matches the  
 11 TransactionID field of the associated KeyRequest message, the access terminal shall  
 12 perform the following in the order in which the requirements are specified:

- 13 • The access terminal shall generate a MessageIntegrityCode equal to the 128 most
- 14 significant bits of  $\text{ehmacsha256}(\text{key}=\text{MICKey}[i], \text{key\_length}=\text{length of MICKey}[i].$
- 15  $\text{message}=\text{Message}, \text{message\_length}=\text{length of Message in units of bits},$
- 16  $\text{message\_offset}=0, \text{MAC\_length}=10)$ , where *Message* is the received ANKeyComplete
- 17 message with the MessageIntegrityCode field set to zero, *i* is the SessionKeyIndex field
- 18 of the corresponding KeyRequest message, and the ehmacsha256 function is specified
- 19 in [2].
- 20 • If the MessageIntegrityCode computed in the previous step does not match the
- 21 MessageIntegrityCode field of ANKeyComplete message, then the access terminal shall
- 22 declare failure, discard the ANKeyComplete message, and stop performing the rest of
- 23 the key exchange procedure.
- 24 • If the Result field of the ANKeyComplete message is not 0x00, the access terminal shall
- 25 declare failure, and stop performing the rest of the key exchange procedures.
- 26 • The access terminal shall set SKey[*i*] to SKeyTemp[*i*], and shall generate corresponding
- 27 Authentication Key and Encryption Key as defined in 2.6.1.4 for values of *i* from
- 28 SessionKeyIndex to  $(\text{SessionKeyIndex} + \text{NumSessionKeys})$ , where SessionKeyIndex and
- 29 NumSessionKeys are fields of the corresponding KeyRequest message and value of *i* is
- 30 calculated modulo 8.

#### 31 2.6.1.2 Access Network Requirements

32 The access network shall initiate the key exchange procedure by sending a KeyRequest  
 33 message. The access network shall set ANNonce to a 128-bit random number.

34 After receiving a KeyResponse message with a TransactionID field that matches the  
 35 TransactionID field of the associated KeyRequest message, the access network shall  
 36 perform the following:

- 37 • If the Result field of the KeyResponse message is not 0x00, then the access network
- 38 shall declare failure and stop performing the rest of the key exchange procedure.

- 1 • The access network shall compute SKeyTemp[*i*], the temporary session key as specified  
2 in 2.6.1.3.
- 3 • The access network shall generate MICKey as specified in 2.6.1.4.
- 4 • The access network shall generate a MessageIntegrityCode equal to the 128 most  
5 significant bits of ehmacsha256(key=MICKey[*i*], key\_length=16, message=Message,  
6 message\_length=length of Message in units of bits, message\_offset=0, MAC\_length=10),  
7 where Message is the received KeyResponse message with the MessageIntegrityCode  
8 field set to zero, *i* is the SessionKeyIndex field of the corresponding KeyRequest  
9 message, and the ehmacsha256 function is specified in [2].
- 10 • If the MessageIntegrityCode computed in the previous step does not match the  
11 MessageIntegrityCode field of KeyResponse message, or if the supported protocol  
12 subtypes sent by the access terminal in the KeyResponse message contain a protocol  
13 (or set of protocols) that the access network supports and prefers to use over the  
14 subtypes that have been negotiated, then the access network shall declare failure, send  
15 an ANKeyComplete message with the appropriate Result, and stop performing the rest  
16 of the key exchange procedure. Otherwise, the access network shall send an  
17 ANKeyComplete message.
- 18 • The access network shall set SKey[*i*] to SKeyTemp[*i*] and generate corresponding  
19 Authentication Key and Encryption Key as specified in 2.6.1.4 for values of *i* from  
20 SessionKeyIndex to (SessionKeyIndex + NumSessionKeys), where SessionKeyIndex and  
21 NumSessionKeys are fields of the corresponding KeyRequest message and value of *i* is  
22 calculated modulo 8.

### 23 2.6.1.3 Session Key Derivation

24 The access terminal and the access network shall derive SKeyTemp[*i*] as follows, where *i* is  
25 the SessionKeyIndex field of the corresponding KeyRequest message:

- 26 • Set *k* and *m* to an 8-bit number with value zero.
- 27 • while  $k < (\text{NumSessionKeys} + 1)$ , where NumSessionKeys is value of the corresponding  
28 field of the KeyRequest message
- 29     – Set SKeyTemp[*i+k*] to  $N_{\text{MKEPSessionKeyLen}}$  least significant bits of { SKeyTemp[*i+k*]  
30 ehmacsha256(key=PairwiseMasterKey, key\_length=length of PairwiseMasterKey in  
31 units of octets, message=ATNonce|ANNonce|*m*, message\_length= length of  
32 message in units of bits, message\_offset=0, MAC\_length=16) }, where the  
33 ehmacsha256 function is specified in [2], and *m* is represented as an 8-bit field.
- 34     – Set *m* to *m*+1
- 35     – Set *k* to *k*+1.

### 36 2.6.1.4 MICKey, Authentication Key and Encryption Key Generation

37 The keys used for MessageIntegrityCode, authentication and encryption are generated from  
38 the session key using the procedures specified in this section.

1 The access network and the access terminal shall compute and store a MICKey,  
 2 Authentication Key, and Encryption Key, derived from each session key. The keys derived  
 3 from SKey[*i*] or SKeyTemp[*i*] are referred to by the subscript *i*. The access network and the  
 4 access terminal shall use the Authentication Key and Encryption Key derived from the  
 5 SKey with index *i*, where *i* is the value of the InUseSessionKeyIndex attribute.

6 The access terminal and the access network shall derive the MICKey[*i*] as follows, where *i* is  
 7 the SessionKeyIndex field of the corresponding KeyRequest message:

- 8 • Set MICKey[*i*] to the 128 most significant bits of {  
 9 ehmacsha256(key=PairwiseMasterKey, key\_length=length of PairwiseMasterKey in  
 10 units of octets, message=ATNonce | ANNonce, message\_length= length of message in  
 11 units of bits, message\_offset=0, MAC\_length=16) }, where the ehmacsha256 function  
 12 is specified in [2].

13 The access network and the access terminal shall set FACAuthKey[*i*], FPCAuthKey[*i*],  
 14 RACAuthKey[*i*], and RPCAuthKey[*i*] to SKey[*i*][127:0], where *i* is the session key index.

15 The access network and the access terminal shall set FACEncKey[*i*], FPCEncKey[*i*],  
 16 RACEncKey[*i*], and RPCEncKey[*i*] to SKey[*i*][255:128], where *i* is the session key index.

## 17 2.6.2 Message Formats

### 18 2.6.2.1 KeyRequest

19 The access network sends the KeyRequest message to initiate the session key exchange.  
 20

Field	Length (bits)
MessageID	8
TransactionID	8
SessionKeyIndex	3
NumSessionKeys	3
Reserved	2
PairwiseMasterKeyID	128
ANNonce	128

21 MessageID The access network shall set this field to 0x00.

22 TransactionID The access network shall set this field to a unique value not being  
 23 used as TransactionID in any other on-going Key Exchange with the  
 24 access terminal.

25 SessionKeyIndex The access network shall set this field to the ID of the SKey for which  
 26 this key exchange is being initiated.

- 1 NumSessionKeys The access network shall set this field to one less than the desired
- 2 number of Session Keys to be generated in this key exchange
- 3 procedure.
  
- 4 Reserved The access network shall set this field to '00'. The access terminal
- 5 shall ignore this field.
  
- 6 PairwiseMasterKeyID
- 7 The access network shall set this field to the 128 most significant bits
- 8 of ehmacsha256(key=PairwiseMasterKey, key\_length= length of
- 9 PairwiseMasterKey in units of octets, message=
- 10 "PairwiseMasterKeyID", message\_length= length of message in units
- 11 of bits, message\_offset=0, MAC\_length=16), where
- 12 PairwiseMasterKeyID is a field of the received KeyRequest message,
- 13 "PairwiseMasterKeyID" is the ASCII encoded value of the string, and
- 14 the ehmacsha256 function is specified in [2].
  
- 15 ANNonce The access network shall set this field to the nonce chosen by the
- 16 access network.

17

<b>Channels</b>	FTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

18 2.6.2.2 KeyResponse

- 19 The access terminal sends the KeyResponse message in response to the KeyRequest
- 20 message.
- 21

<b>Field</b>	<b>Length (bits)</b>
MessageID	8
TransactionID	8
Result	8
ATNonce	0 or 128
SecuritySubtypeCount	0 or 8

Zero or SecuritySubtypeCount occurrences of the following field:

SecuritySubtype	16
-----------------	----

AuthenticationSubtypeCount	0 or 8
AuthenticationSubtypeCount occurrences of the following field:	
AuthenticationSubtype	16

EncryptionSubtypeCount	0 or 8
EncryptionSubtypeCount occurrences of the following field:	
EncryptionSubtype	16

KeyExchangeSubtypeCount	0 or 8
KeyExchangeSubtypeCount occurrences of the following field:	
KeyExchangeSubtype	16

MessageIntegrityCode	0 or 80
----------------------	---------

- |   |               |  |
|---|---------------|--|
| 1 | MessageID     | The access terminal shall set this field to 0x01.                      |
| 2 | TransactionID | The access terminal shall set this field to the value of the           |
| 3 |               | TransactionID field of the KeyRequest message to which the access      |
| 4 |               | terminal is responding.  |
| 5 | Result        | The access terminal shall set this field according to Table 2.6.2.2-1. |

1

**Table 2.6.2.2-1. Definition of Result field**

<b>Value</b>	<b>Meaning</b>
0x00	Continue key exchange.
0x03	PairwiseMasterKey not found.
All other values	Reserved

2 ATNonce

If Result is 0x00, then the access terminal shall set this field to the nonce chosen by the access terminal. Otherwise, the access terminal shall omit this field.

3

5 SecuritySubtypeCount

If Result is 0x00, then the access terminal shall set this field to the number of subtypes of the Security Protocol supported by the access terminal besides the HardLink subtype and besides the default subtype. Otherwise, the access terminal shall omit this field.

6

7

8

9

10 SecuritySubtype

If Result is 0x00, then the access terminal shall set this field to the subtype of the Security Protocol supported by the access terminal. Otherwise, the access terminal shall omit this field. The access terminal shall not set this field to 0xfffe (HardLink subtype). The access terminal shall not set this field to 0x0000.

11

12

13

14

15 AuthenticationSubtypeCount

If Result is 0x00, then the access terminal shall set this field to the number of subtypes of the Authentication Protocol supported by the access terminal besides the HardLink subtype and besides the default subtype. Otherwise, the access terminal shall omit this field.

16

17

18

19

20 AuthenticationSubtype

If Result is 0x00, then the access terminal shall set this field to the subtype of the Authentication Protocol supported by the access terminal. Otherwise, the access terminal shall omit this field. The access terminal shall not set this field to 0xfffe (HardLink subtype). The access terminal shall not set this field to 0x0000.

21

22

23

24

25

26 EncryptionSubtypeCount

If Result is 0x00, then the access terminal shall set this field to the number of subtypes of the Encryption Protocol supported by the access terminal besides the HardLink subtype and besides the default subtype. Otherwise, the access terminal shall omit this field.

27

28

29

30

31 EncryptionSubtype

If Result is 0x00, then the access terminal shall set this field to the subtype of the Encryption Protocol supported by the access terminal. Otherwise, the access terminal shall omit this field. The access

32

33

1 terminal shall not set this field to 0xfffe (HardLink subtype). The  
2 access terminal shall not set this field to 0x0000.

### 3 KeyExchangeSubtypeCount

4 If Result is 0x00, then the access terminal shall set this field to the  
5 number of subtypes of the Key Exchange Protocol supported by the  
6 access terminal besides the HardLink subtype and besides the  
7 default subtype. Otherwise, the access terminal shall omit this field.

### 8 KeyExchangeSubtype

9 If Result is 0x00, then the access terminal shall set this field to the  
10 subtype of the Key Exchange Protocol supported by the access  
11 terminal. Otherwise, the access terminal shall omit this field. The  
12 access terminal shall not set this field to 0xfffe (HardLink subtype).  
13 The access terminal shall not set this field to 0x0000.

### 14 MessageIntegrityCode

15 If Result is 0x00, then the access terminal shall set this field to the  
16 128 most significant bits of ehmacsha256(key=MICKey[*i*],  
17 key\_length= length of MICKey[*i*] in units of octets, message=*Message*,  
18 message\_length= length of *Message* in units of bits,  
19 message\_offset=0, MAC\_length=10), where *Message* is set to all fields  
20 of this message with this field set to zero, *i* is SessionKeyIndex field of  
21 the corresponding KeyRequest message, and the ehmacsha256  
22 function is specified in [2]. Otherwise, the access terminal shall omit  
23 this field.  
24

<b>Channels</b>	RTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

### 25 2.6.2.3 ANKeyComplete

26 The access network sends the ANKeyComplete message in response to the KeyResponse  
27 message.  
28

Field	Length (bits)
MessageID	8
TransactionID	8
Result	8
MessageIntegrityCode	80

29 MessageID The access network shall set this field to 0x02.

30 TransactionID The access network shall set this field to the value of the  
31 TransactionID field of the corresponding KeyResponse message.

1 Result The access network shall set this field according to Table 2.6.2.3-1.

2 **Table 2.6.2.3-1. Definition of Result field**

Value	Meaning
0x00	Key exchange successful.
0x01	MessageIntegrityCode failed.
0x02	MessageIntegrityCode successful, but security capabilities verification failed.
All other values	Reserved

3 MessageIntegrityCode

4 The access network shall set this field to 128 most significant bits of  
 5 ehmacsha256(key=MICKey[i], key\_length= length of MICKey[i] in  
 6 units of octets, message=Message, message\_length= length of  
 7 Message in units of bits, message\_offset=0, MAC\_length=10), where  
 8 Message is set to all fields of this message with this field set to zero, i  
 9 is the SessionKeyIndex field of the corresponding KeyRequest  
 10 message, and the ehmacsha256 function is specified in [2].  
 11

<b>Channels</b>	FTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

12 2.6.2.4 AttributeUpdateRequest

13 The sender sends an AttributeUpdateRequest message to offer a set of attribute values for a  
 14 given attribute.  
 15

Field	Length (bits)
MessageID	8
TransactionID	8

One or more instances of the following record

AttributeRecord	Attribute dependent
-----------------	---------------------

16 MessageID The sender shall set this field to 0x52.

17 TransactionID The sender shall increment this value for each new  
 18 AttributeUpdateRequest message sent.

19 AttributeRecord The format of this record is specified in [1].  
 20

<b>Channels</b>	CC	FTC	RTC	<b>SLP</b>	Reliable on FTC and RTC Best Effort on CC
<b>Addressing</b>	unicast			<b>Priority</b>	40

## 1 2.6.2.5 AttributeUpdateAccept

2 The sender sends an AttributeUpdateAccept message in response to an  
3 AttributeUpdateRequest message to select an attribute value from a list of offered values.

4

Field	Length (bits)
MessageID	8
TransactionID	8

5 MessageID The sender shall set this field to 0x53.

6 TransactionID The sender shall set this value to the TransactionID field of the  
7 corresponding AttributeUpdateRequest message.

8

<b>Channels</b>	AC	FTC	RTC	<b>SLP</b>	Reliable on FTC and RTC Best Effort on AC
<b>Addressing</b>	unicast			<b>Priority</b>	40

## 9 2.6.2.6 AttributeUpdateReject

10 The access network sends an AttributeUpdateReject message to reject the set of attribute  
11 values proposed by the access terminal in the corresponding AttributeUpdateRequest  
12 message.

13

Field	Length (bits)
MessageID	8
TransactionID	8

14 MessageID The access network shall set this field to 0x54.

15 TransactionID The access network shall set this value to the TransactionID field of  
16 the corresponding AttributeUpdateRequest message.

17

<b>Channels</b>	FTC	<b>SLP</b>	Reliable
<b>Addressing</b>	unicast	<b>Priority</b>	40

1 2.6.3 Interface to Other Protocols

2 2.6.3.1 Commands

3 This protocol does not issue any commands.

4 2.6.3.2 Indications

5 This protocol does not register to receive any indications.

6 **2.7 Configuration Attributes**

7 The configurable simple attribute for this protocol is listed in Table 2.7-1.

8 The access terminal shall use as defaults the values in Table 2.7-1 that are typed in ***bold italics***.

10 Unless specified otherwise, the access terminal and the access network shall not use the  
 11 Generic Attribute Update Protocol to update configurable attributes belonging to the Multi-  
 12 Key Key Exchange Protocol. The access terminal and the access network shall support the  
 13 use of the Generic Attribute Update Protocol to update values of the InUseSessionKeyIndex  
 14 attribute. The access terminal shall not send an AttributeUpdateRequest message  
 15 containing the InUseSessionKeyIndex attribute. The access network may include the  
 16 InUseSessionKeyIndex attribute in an AttributeUpdateRequest message sent on the Control  
 17 Channel.

18

**Table 2.7-1. Configurable Values**

<b>Attribute ID</b>	<b>Attribute</b>	<b>Values</b>	<b>Meaning</b>
0x00	InUseSessionKeyIndex	<b>0x00</b>	SKey[0] is used.
		0x01	SKey[1] is used
		0x02	SKey[2] is used
		0x03	SKey[3] is used
		0x04	SKey[4] is used
		0x05	SKey[5] is used
		0x06	SKey[6] is used
		0x07	SKey[7] is used
		0x08-0xff	Reserved

1 **2.8 Protocol Numeric Constants**

Constant	Meaning	Value
N <sub>KEPType</sub>	Type field for this protocol	See [3]
N <sub>MKEP</sub>	Subtype field for this protocol	0x0003
N <sub>MKEPSessionKeyLen</sub>	Length of Session Key in units of bits	256

2 **2.9 Session State Information**

3 The Session State Information record (see [1]) consists of parameter records.

4 This protocol defines the following parameter record in addition to the configuration  
5 attributes for this protocol.

## 6 2.9.1 SKey Parameter

7 **Table 2.9.1-1. The Format of the Parameter Record for the Skey Parameter**

Field	Length (bits)
ParameterType	8
Length	8
SKey[0]Included	1
SKey[1]Included	1
SKey[2]Included	1
SKey[3]Included	1
SKey[4]Included	1
SKey[5]Included	1
SKey[6]Included	1
SKey[7]Included	1
SKey[0]	0 or 128
SKey[1]	0 or 128
SKey[2]	0 or 128
SKey[3]	0 or 128
SKey[4]	0 or 128
SKey[5]	0 or 128
SKey[6]	0 or 128
SKey[7]	0 or 128

8 ParameterType This field shall be set to 0x01 for this parameter record.

9 Length This field shall be set to the length of this parameter record in units  
10 of octets excluding the Length field.

1	SKey[0]Included	If SKey[0] is zero, then this field shall be set to '0'. Otherwise, this
2		field shall be set to '1'.
3	SKey[1]Included	If SKey[1] is zero, then this field shall be set to '0'. Otherwise, this
4		field shall be set to '1'.
5	SKey[2]Included	If SKey[2] is zero, then this field shall be set to '0'. Otherwise, this
6		field shall be set to '1'.
7	SKey[3]Included	If SKey[3] is zero, then this field shall be set to '0'. Otherwise, this
8		field shall be set to '1'.
9	SKey[4]Included	If SKey[4] is zero, then this field shall be set to '0'. Otherwise, this
10		field shall be set to '1'.
11	SKey[5]Included	If SKey[5] is zero, then this field shall be set to '0'. Otherwise, this
12		field shall be set to '1'.
13	SKey[6]Included	If SKey[6] is zero, then this field shall be set to '0'. Otherwise, this
14		field shall be set to '1'.
15	SKey[7]Included	If SKey[7] is zero, then this field shall be set to '0'. Otherwise, this
16		field shall be set to '1'.
17	SKey[0]	If SKey[0]Included is '0', then this field shall be omitted. Otherwise,
18		this field shall be set to the value of the session key with key index
19		0x00.
20	SKey[1]	If SKey[1]Included is '0', then this field shall be omitted. Otherwise,
21		this field shall be set to the value of the session key with key index
22		0x01.
23	SKey[2]	If SKey[2]Included is '0', then this field shall be omitted. Otherwise,
24		this field shall be set to the value of the session key with key index
25		0x02.
26	SKey[3]	If SKey[3]Included is '0', then this field shall be omitted. Otherwise,
27		this field shall be set to the value of the session key with key index
28		0x03.
29	SKey[4]	If SKey[4]Included is '0', then this field shall be omitted. Otherwise,
30		this field shall be set to the value of the session key with key index
31		0x04.
32	SKey[5]	If SKey[5]Included is '0', then this field shall be omitted. Otherwise,
33		this field shall be set to the value of the session key with key index
34		0x05.

- 1 SKey[6] If SKey[6]Included is '0', then this field shall be omitted. Otherwise,  
 2 this field shall be set to the value of the session key with key index  
 3 0x06.
- 4 SKey[7] If SKey[7]Included is '0', then this field shall be omitted. Otherwise,  
 5 this field shall be set to the value of the session key with key index  
 6 0x07.

7 2.9.2 PMK Parameter

8 **Table 2.9.2-1. The Format of the Parameter Record for the PMK Parameter**

Field	Length (bits)
ParameterType	8
Length	8
PMKCount	8

PMKCount occurrences of the following three fields:

PMKLength	8
PMK	PMKLength × 8
PMKExpirytime	32

- 9 ParameterType This field shall be set to 0x04 for this parameter record.
- 10 Length This field shall be set to the length of this parameter record in units  
 11 of octets excluding the Length field.
- 12 PMKCount This field shall be set to the number of occurrences of the PMK field  
 13 in this parameter record.
- 14 PMKLength This field shall be set to the length of the PMK field in units of octets.
- 15 PMK This field shall be set to a PairwiseMasterKey.
- 16 PMKExpirytime This field shall be set to the expiry time, in units of seconds, of the  
 17 PairwiseMasterKey.

18