

**3GPP2 C.S0039-0**

**Version 2.0**

**May, 2006**



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

---

## ***Enhanced Subscriber Privacy for cdma2000 High Rate Packet Data***

**© 3GPP2 2006**

*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.*



**CONTENTS**

1	1. Overview .....	1-1
2	2. Time-Counter-Based Security Protocol .....	2-1
3	2.1. Overview.....	2-1
4	2.2. Primitives and Public Data .....	2-1
5	2.2.1. Commands .....	2-1
6	2.2.2. Return Indications .....	2-1
7	2.2.3. Public Data.....	2-1
8	2.3. Protocol Data Unit.....	2-1
9	2.4. Protocol Initialization.....	2-1
10	2.4.1. Protocol Initialization for the InConfiguration Protocol Instance .....	2-1
11	2.5. Procedures and Messages for the InConfiguration Instance of the Protocol .....	2-2
12	2.5.1. Procedures.....	2-2
13	2.5.2. Commit Procedures .....	2-2
14	2.5.3. Message Formats .....	2-3
15	2.5.3.1. ConfigurationRequest .....	2-3
16	2.5.3.2. ConfigurationResponse .....	2-3
17	2.6. Procedures and Messages for the InUse Instance of the Protocol.....	2-4
18	2.6.1. Procedures.....	2-4
19	2.6.1.1. Transmit Procedures.....	2-4
20	2.6.1.1.1. Generation of the Cryptosync .....	2-4
21	2.6.1.1.2. Construction of the Security Protocol Header .....	2-5
22	2.6.1.2. Receive Procedures .....	2-6
23	2.6.2. Message Formats .....	2-8
24	2.6.3. Time-Counter-Based Security Protocol Header.....	2-8
25	2.6.4. Time-Counter-Based Security Protocol Trailer.....	2-8
26	2.6.5. Interface to Other Protocols .....	2-8
27	2.6.5.1. Commands .....	2-8
28	2.6.5.2. Indications.....	2-8
29	2.7. Configuration Attributes.....	2-9
30	2.7.1. FTC Cryptosync Attribute .....	2-9
31	2.7.2. RTC Cryptosync Attribute .....	2-10
32	2.7.3. CC Cryptosync Attribute .....	2-12
33	2.7.4. AC Cryptosync Attribute .....	2-14
34	2.8. Protocol Numeric Constants .....	2-15
35	2.9. Session State Information.....	2-15
36	2.9.1. KeyIndex Parameter.....	2-16
37	2.9.2. TimeStampLong Parameter .....	2-16
38	2.9.3. Counter Parameter .....	2-17

1	3. AES Encryption Protocol .....	3-1
2	3.1. Primitives and Public Data .....	3-1
3	3.1.1. Commands .....	3-1
4	3.1.2. Return Indications.....	3-1
5	3.1.3. Public Data .....	3-1
6	3.2. Protocol Data Unit.....	3-1
7	3.3. Protocol Initialization .....	3-1
8	3.3.1. Protocol Initialization for the InConfiguration Protocol Instance.....	3-1
9	3.4. Procedures and Messages for the InConfiguration Instance of the Protocol.....	3-2
10	3.4.1. Procedures .....	3-2
11	3.4.2. Commit Procedures .....	3-2
12	3.4.3. Message Formats.....	3-2
13	3.4.3.1. ConfigurationRequest .....	3-2
14	3.4.3.2. ConfigurationResponse.....	3-3
15	3.5. Procedures and Messages for the InUse Instance of the Protocol .....	3-3
16	3.5.1. Procedures .....	3-3
17	3.5.1.1. Constructing the Encryption Key .....	3-3
18	3.5.1.2. Constructing the Cryptosync .....	3-6
19	3.5.1.3. Transmit Procedures .....	3-6
20	3.5.1.4. Receive Procedures .....	3-7
21	3.5.2. Message Formats.....	3-8
22	3.5.3. AES Encryption Protocol Header.....	3-8
23	3.5.4. AES Encryption Protocol Trailer .....	3-8
24	3.5.5. Interface to Other Protocols .....	3-8
25	3.5.5.1. Commands .....	3-8
26	3.5.5.2. Indications .....	3-9
27	3.6. Configuration Attributes .....	3-9
28	3.6.1. FTCReducedStrengthEncryptionKey Attribute .....	3-10
29	3.6.2. RTCReducedStrengthEncryptionKey Attribute .....	3-11
30	3.6.3. CCReducedStrengthEncryptionKey Attribute.....	3-11
31	3.6.4. ACReducedStrengthEncryptionKey Attribute .....	3-12
32	3.7. Protocol Numeric Constants .....	3-13
33	3.8. Session State Information .....	3-13

**FIGURES**

1 Figure 1-1. Security Layer Encapsulation..... 1-1  
2

**TABLES**

1	Table 2.6.1-1. Subfield of the Cryptosync.....	2-4
2	Table 2.6.1-2. Encoding of the ChannelID Field .....	2-4
3	Table 2.6.1-3. The Format of the Security Protocol Header .....	2-6
4	Table 2.6.1-4. Subfield of the Cryptosync.....	2-7
5	Table 2.6.1-5. Encoding of the ChannelID Field .....	2-7
6	Table 2.9.1-1. The Format of the Parameter Record for the KeyIndex Parameter.....	2-16
7	Table 2.9.2-1. The Format of the Parameter Record for the TimeStampLong	
8	Parameter .....	2-16
9	Table 2.9.2-2. Encoding of the ParameterType Field.....	2-17
10	Table 2.9.3-1. The Format of the Parameter Record for the Counter Parameter .....	2-17
11	Table 2.9.3-2. Encoding of the ParameterType Field.....	2-18
12	Table 3.5.5-1. FTCEncryption .....	3-9
13	Table 3.5.5-2. RTCEncryption.....	3-9
14	Table 3.5.5-3. CCEncryption.....	3-9
15	Table 3.5.5-4. ACEncryption.....	3-10

1 **(This foreword is not part of this Standard)**

2 This standard was prepared by Technical Specification Group C of the Third Generation  
3 Partnership Project 2 (3GPP2). This standard is evolved from and is a companion to the  
4 cdma2000<sup>®1</sup> standards. This air interface standard provides Enhanced Subscriber Privacy  
5 requirements for cdma2000 High Rate Packet Data.

6

---

<sup>1</sup> “cdma2000<sup>®</sup> 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<sup>®</sup> is a registered trademark of the Telecommunications Industry Association (TIA-USA) in the United States.”

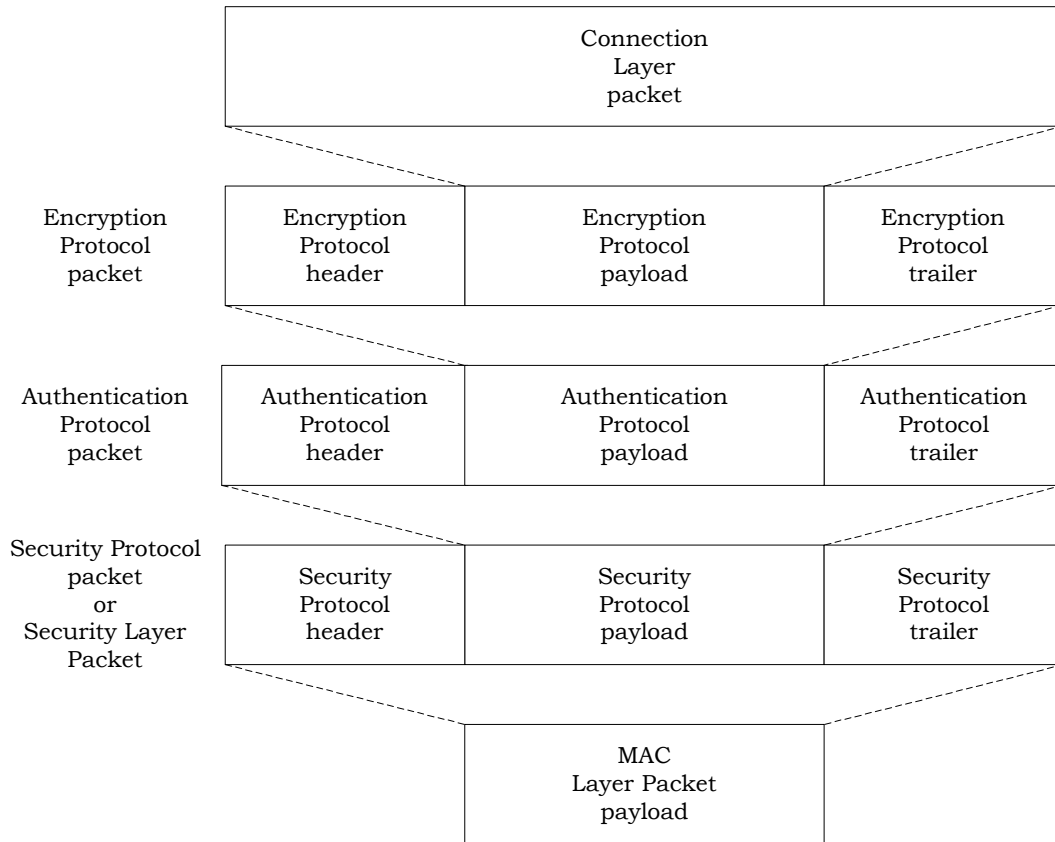
## REFERENCES

1 The following standards contain provisions, which, through reference in this text,  
2 constitute provisions of this standard. At the time of publication, the editions indicated  
3 were valid. All standards are subject to revision, and parties to agreements based on this  
4 standard are encouraged to investigate the possibility of applying the most recent editions  
5 of the standards indicated below.

- 6  
7 [1] 3GPP2 C.S00024, cdma2000 High Rate Packet Data Air Interface Specification.  
8 [2] [3GPP2 S.S0055-A, Version 3.0, Enhanced Cryptographic Algorithms, October,](#)  
9 [2005](#) ~~TR45.AHAG, Enhanced Cryptographic Algorithms, Revision B, March 5, 2002~~  
10 [3] [3GPP2 S.S0078-A, Version 3.0, Common Security Algorithms, October, 2005](#)

1 **1. OVERVIEW**

2 Figure 1-1 shows the relationship between a Connection Layer packet, an Encryption  
 3 Protocol packet, an Authentication Protocol packet, a Security Protocol packet, and a MAC  
 4 Layer packet payload.



5  
6

**Figure 1-1. Security Layer Encapsulation**

7 When a Connection Layer packet that is to be authenticated or encrypted is delivered to the  
 8 Security Layer, the following steps are performed by the protocols in the Security Layer in  
 9 the order specified below:

- 10 • The Security Layer protocol generates a ~~cryptosync~~ Cryptosync for the channel for  
 11 which the Connection Layer packet is destined. For the purpose for referencing this  
 12 value of ~~cryptosync~~ Cryptosync in the following steps, denote this value as  
 13 TheCryptosync.
- 14 • The Connection Layer packet and TheCryptosync are delivered to the Encryption  
 15 Protocol.
- 16 • If the Connection Layer packet is to be encrypted, the Encryption Protocol uses  
 17 TheCryptosync, the encryption key, and other parameters specified by the  
 18 Encryption Protocol (if any) to encrypt the Connection Layer packet and construct  
 19 the Encryption Protocol packet.

- 1 • The Encryption Protocol delivers the Encryption Protocol packet and TheCryptosync  
2 to the Authentication Protocol.
- 3 • If the Encryption Protocol packet is to be authenticated, the Authentication Protocol  
4 uses TheCryptosync, authentication key, and other parameters specified by the  
5 Authentication Protocol to construct the Authentication Protocol packet.
- 6 • The Authentication Protocol delivers the Authentication Protocol packet and  
7 TheCryptosync to the Security Protocol.
- 8 • The Security Protocol uses TheCryptosync to construct the Security Protocol header  
9 and trailer (if any).
- 10 • The Security Protocol delivers the Security Protocol packet to the MAC layer.

11 Conversely, when the Security Layer receives a MAC Layer Packet payload that is either  
12 authenticated or encrypted, the following steps are performed by the protocols in the  
13 Security Layer in the order specified below:

- 14 • The Security Protocol constructs the Cryptosync using the Security Protocol header  
15 and trailer (if any). For the purpose for referencing this value of ~~cryptosync~~  
16 [Cryptosync](#) in the following steps, denote this value as “TheCryptosync”.
- 17 • The Security Protocol removes the Security Protocols header and trailer and delivers  
18 TheCryptosync and the Security Protocol payload to the Authentication Protocol.
- 19 • If the Authentication Protocol packet is authenticated, the Authentication Protocol  
20 uses TheCryptosync, authentication key, Authentication Protocol payload,  
21 Authentication Protocol header and trailer, and other parameters specified by the  
22 Authentication Protocol (if any) to verify the authentication signature. If the  
23 authentication signature passes, then the Authentication Protocol delivers the  
24 Authentication Protocol payload to the Encryption Protocol, otherwise the  
25 Authentication Protocol Packet is discarded. ~~If the authentication signature does  
26 not pass, then the Authentication Protocol discards the packet.~~
- 27 • If the Authentication Protocol packet is not authenticated, then the Authentication  
28 Protocol delivers the Authentication Protocol payload to the Encryption Protocol.
- 29 • If the Encryption Protocol packet is encrypted, the Encryption Protocol uses  
30 TheCryptosync and the encryption key to decrypt the Encryption Protocol packet.  
31 The decrypted payload is then delivered to the Connection Layer.
- 32 • If the Encryption Protocol packet is not encrypted, the Encryption Protocol packet is  
33 delivered to the Connection Layer.
- 34 • The Security Layer provides two pairs of security information to the Connection  
35 Layer. The first indicates:
  - 36 – Whether or not the Security Layer session configuration supported encryption of  
37 the Security Layer packet, and

1           – Whether or not the Security Layer decrypted the Security Layer packet<sup>2</sup>.

2           The second indicates:

3           – Whether or not the Security Layer session configuration supported  
4           authentication ~~encryption~~ of the Security Layer packet, and

5           – Whether or not the Security Layer authenticated the Security Layer packet.

6           The receiving application or protocol may use these two pairs of security information  
7           to determine whether or not to discard the payload.

8           The access terminal shall not require Connection Layer packets that satisfy any of the  
9           following conditions to be encrypted:

- 10           • A Connection Layer packet that is received on the Control Channel and its  
11           encapsulating MAC Layer packets was not addressed using the Unicast Addressing  
12           mode.
- 13           • A Connection Layer packet that is received on the Control Channel and contains a  
14           SessionClose<sup>3</sup> message associated with the Default Session Management Protocol.
- 15           • The Connection Layer packets that contain any of the messages that are excluded  
16           from being encrypted by the protocol that defines the message.

---

<sup>2</sup> For example, a Security Layer packet that is received on the Control Channel is decrypted by the AES Encryption protocol if the configured value of the CCEncryption attribute is 0x01 and the SecurityLayerFormat bit of the encapsulating MAC Layer packet is equal to '1'.

<sup>3</sup> The access network must be able to close the session in case it does not have access terminals session (e.g., if it cannot retrieve the session from the old subnet).

1 No text.

## 1 **2. TIME-COUNTER-BASED SECURITY PROTOCOL**

### 2 **2.1. Overview**

3 The Time-Counter-Based Security protocol performs the following tasks:

- 4 • On the transmission side, this protocol provides a ~~cryptosyne~~[Cryptosync](#) that may  
5 be used by the negotiated Authentication Protocol and Encryption Protocol.
- 6 • On the receiving side, this protocol computes the ~~cryptosyne~~[Cryptosync](#) using the  
7 information provided in the Security Protocol header and makes the ~~cryptosyne~~  
8 [Cryptosync](#) publicly available.

### 9 **2.2. Primitives and Public Data**

#### 10 2.2.1. Commands

11 This protocol does not define any commands.

#### 12 2.2.2. Return Indications

13 This protocol does not return any indications.

#### 14 2.2.3. Public Data

15 This protocol shall make the following data public:

- 16 • Subtype for this protocol
- 17 • Cryptosync and CryptosyncLength for Security Layer packets associated with the  
18 FTC
- 19 • Cryptosync and CryptosyncLength for Security Layer packets associated with the  
20 RTC
- 21 • Cryptosync and CryptosyncLength for Security Layer packets associated with the CC
- 22 • Cryptosync and CryptosyncLength for Security Layer packets associated with the AC

### 23 **2.3. Protocol Data Unit**

24 The protocol data unit for this protocol is a Security Layer packet.

### 25 **2.4. Protocol Initialization**

#### 26 2.4.1. Protocol Initialization for the InConfiguration Protocol Instance

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

- 29 • The fall-back values of the attributes for this protocol instance shall be set to the  
30 default values specified for each attribute.

- 1 • If the InUse instance of this protocol has the same protocol subtype as this  
2 InConfiguration protocol instance, then the fall-back values of the attributes defined  
3 by the InConfiguration protocol instance shall be set to the values of the  
4 corresponding attributes associated with the InUse protocol instance.
- 5 • The value for each attribute for this protocol instance shall be set to the fall-back  
6 value for that attribute.

#### 7 ~~2.4.2. Protocol Initialization for the InUse Protocol Instance~~

8 ~~Upon creation of the InUse instance of this protocol, the access terminal and the access~~  
9 ~~network shall set the value of the attributes for this protocol instance to the default values~~  
10 ~~specified for each attribute.~~

### 11 **2.5. Procedures and Messages for the InConfiguration Instance of the Protocol**

#### 12 2.5.1. Procedures

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

#### 15 2.5.2. Commit Procedures

16 The access terminal and the access network shall perform the procedures specified in this  
17 section, in the order specified, when directed by the InUse instance of the Session  
18 Configuration Protocol (see [1]) to execute the Commit procedures:

- 19 • All the public data that are defined by this protocol, but are not defined by the InUse  
20 protocol instance shall be added to the public data of the InUse protocol.
- 21 • If the InUse instance of this protocol has the same subtype as this protocol instance,  
22 then
  - 23 – The access terminal and the access network shall set the attribute values  
24 associated with the InUse instance of this protocol to the attribute values  
25 associated with the InConfiguration instance of this protocol, and
  - 26 – The access terminal and the access network shall purge the InConfiguration  
27 instance of the protocol.
- 28 • If the InUse instance of this protocol does not have the same subtype as this protocol  
29 instance, then the access terminal and the access network shall perform the  
30 following:
  - 31 – The InConfiguration protocol instance shall become the InUse protocol instance  
32 for the Security Protocol.
- 33 • All the public data not defined by this protocol shall be removed from the public data  
34 of the InUse protocol.

1 2.5.3. Message Formats

2 2.5.3.1. ConfigurationRequest

3 The ConfigurationRequest message format is as follows:

4  
5

Field	Length (bits)
MessageID	8
TransactionID	8

Zero or more instances of the following record

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

- 6 MessageID The sender shall set this field to 0x50.
- 7 TransactionID The sender shall increment this value for each new  
8 ConfigurationRequest message sent.
- 9 AttributeRecord The format of this record is specified in [1].

10

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

11 2.5.3.2. ConfigurationResponse

12 The ConfigurationResponse message format is as follows:

13

Field	Length (bits)
MessageID	8
TransactionID	8

Zero or more instances of the following record

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

- 14 MessageID The sender shall set this field to 0x51.
- 15 TransactionID The sender shall set this value to the TransactionID field of the  
16 corresponding ConfigurationRequest message.
- 17 AttributeRecord An attribute record containing a single attribute value. If this  
18 message selects a complex attribute, only the ValueID field of the  
19 complex attribute shall be included in the message. The format of the

1 AttributeRecord is given in [1]. The sender shall not include more  
 2 than one attribute record with the same attribute identifier.  
 3

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

4 **2.6. Procedures and Messages for the InUse Instance of the Protocol**

5 2.6.1. Procedures

6 Each Security Layer packet consists of an Authentication Protocol packet and a Security  
 7 Protocol header.

8 2.6.1.1. Transmit Procedures

9 2.6.1.1.1. Generation of the Cryptosync

10 When the Security Layer receives a Connection Layer packet that is to be either  
 11 authenticated or encrypted, the Security Protocol shall compute the Cryptosync for the  
 12 channel on which the Security Layer packet to be sent as shown in Table 2.6.1-1.

13  
 14

**Table 2.6.1-1. Subfield of the Cryptosync**

<b>Subfield</b>	<b>Length (bits)</b>
ChannelID	8
TimeStampLong	TimeStampLongLength
Counter	CounterLength
<a href="#">BandClass</a>	<a href="#">5</a>
<a href="#">ChannelNumber</a>	<a href="#">11</a>

15 ChannelID This field is encoded as specified in Table 2.6.1-2.

16 **Table 2.6.1-2. Encoding of the ChannelID Field**

<b>ChannelID Value</b>	<b>Meaning</b>
0x00	Cryptosync used for the FTC
0x01	Cryptosync used for the RTC
0x02	Cryptosync used for the CC
0x03	Cryptosync used for the AC
Other values	Reserved

17 TimeStampLong If the TimeStampShortLength for the channel specified by the  
 18 ChannelID is zero, then this field shall be set to the LSBs of the  
 19 CDMA System Time, in units of time specified by TimeStampUnit for

1 the channel specified by the ChannelID, corresponding to the time  
 2 when the Physical Layer will begin transmission of the Security Layer  
 3 Packet. Otherwise, the sender shall set this field to the LSBs of the  
 4 CDMA System Time, in units of time specified by TimeStampUnit for  
 5 the channel specified by the ChannelID, corresponding to a time that  
 6 is not later than when the Physical Layer will begin transmission of  
 7 the Security Layer Packet<sup>4</sup>.

8 Counter This field shall be set to the number of Security Layer packets  
 9 (modulo  $2^{\text{CounterLength}}$ ) that have been generated since time T such that  
 10  $T \bmod \text{TimeStampUnit} = 0$ , where T is the CDMA System Time in  
 11 units of slots.

12 BandClass This field shall be set to the band class number associated with the  
 13 CDMA Channel on which the packet is to be transmitted.

14 ChannelNumber This field shall be set to the channel number associated with the  
 15 CDMA Channel on which the packet is to be transmitted.

16 TimeStampLongLength, CounterLength, and TimeStampShortLength are configurable  
 17 parameters associated with the channel identified by the ChannelID.

#### 18 2.6.1.1.2. Construction of the Security Protocol Header

19 The protocol shall construct a Security Layer packet out of the Authentication Protocol  
 20 packet as follows and shall pass the packets for transmission to the MAC Layer:

- 21 • When the protocol receives an Authentication Protocol packet from the  
 22 Authentication Protocol that is either authenticated or encrypted, it shall set the  
 23 Security Protocol header as shown in Table 2.6.1-3. The Security Protocol shall  
 24 then prepend ~~add~~ the Security Protocol header ~~to in front of~~ the Authentication  
 25 Protocol packet.

---

<sup>4</sup> For example, if the sender knows that there is a fixed delay between the generation of the Security Layer Packets and the time that they are going to be transmitted, then the sender can set the TimeStampLong to the expected transmission time. This can potentially reduce the number of bits that are needed to represent TimeStampShort. The current System Time is a valid choice.

1

**Table 2.6.1-3. The Format of the Security Protocol Header**

Subfield	Length (bits)
KeyIndex	KeyIndexLength
TimeStampShort	TimeStampShortLength
Counter	0 or CounterLength
Reserved	0 to 7 (as required)

2 KeyIndex If the Key Exchange protocol provides the KeyIndex as one of its  
3 public data, then the sender shall set this field to the KeyIndexLength  
4 LSBs of the current value of KeyIndex. Otherwise, the sender shall  
5 set this field to zero. KeyIndexLength is the length of the KeyIndex  
6 and is specified as a configurable parameter for the channel on which  
7 the Security Layer packet is to be sent.

8 TimeStampShort If the TimeStampShortLength is zero, this field shall be omitted.  
9 Otherwise, the sender shall set this field to the LSBs of the  
10 TimeStampLong sub-field of the ~~cryptosync~~ Cryptosync for the  
11 destination channel of the Security Protocol packet.

12 Counter If the CounterExplicit is zero or CounterLength is zero, this field shall  
13 be omitted. Otherwise, the sender shall set this field to the Counter  
14 sub-field of the ~~cryptosync~~ Cryptosync for the destination channel of  
15 the Security Protocol packet.

16 Reserved If included, the sender shall set this field to zero. The length of this  
17 field shall be the smallest number of bits that makes the Security  
18 Protocol Header an integer number of octets.

19 CounterExplicit, CounterLength, KeyIndexLength, and TimeStampShortLength are  
20 configurable parameters associated with the destination channel of the Security  
21 Protocol packet.

22 • When the protocol receives an Authentication Protocol packet from the  
23 Authentication Protocol that is neither authenticated nor encrypted, the protocol  
24 shall not add a security protocol header to the Authentication Protocol packet.

25 • This protocol shall not append a Security Protocol trailer to the Authentication  
26 Protocol packet.

#### 27 2.6.1.2. Receive Procedures

28 ~~This~~ The Security Protocol shall construct the Authentication Protocol packet using the  
29 Security Layer packet (received from the MAC Layer) as follows and shall forward the  
30 packet to the Authentication Protocol:

- 1 • When the protocol receives a Security Layer packet from the MAC Layer that is either  
2 authenticated or encrypted, it shall construct the Authentication Protocol packet by  
3 removing the Security Layer header.
- 4 • When the protocol receives a Security Layer packet from the MAC Layer that is  
5 neither authenticated nor encrypted, it shall set the Authentication Protocol packet  
6 to the Security Layer packet.

7 When the Security Protocol receives a Security Layer packet from the MAC Layer that is  
8 either authenticated or encrypted, it shall compute the Cryptosync for the channel on  
9 which the Security Layer packet is received as shown in Table 2.6.1-4. The Security  
10 Protocol shall deliver the Authentication Protocol packet together with the computed value  
11 of the ~~cryptosyne~~ Cryptosync to the Authentication Protocol.

12  
13 **Table 2.6.1-4. Subfield of the Cryptosync**

Subfield	Length (bits)
ChannelID	8
TimeStamPLong	TimeStamPLongLength
Counter	CounterLength
<u>BandClass</u>	<u>5</u>
<u>ChannelNumber</u>	<u>11</u>

14 ChannelID This field shall be set according to Table 2.6.1-5.

15 **Table 2.6.1-5. Encoding of the ChannelID Field**

ChannelID Value	Meaning
0x00	The Security Layer packet is associated with the FTC
0x01	The Security Layer packet is associated with the RTC
0x02	The Security Layer packet is associated with the CC
0x03	The Security Layer packet is associated with the AC
Other values	Reserved

16 TimeStamPLong If the TimeStampShortLength for the channel specified by the  
17 ChannelID is zero, then this field shall be set to the LSBs of the  
18 system time corresponding to the time when the Security Layer  
19 Packet is received in units of time specified by TimeStampUnit for the  
20 channel specified by the ChannelID.

21 Otherwise, this field shall be derived from the TimeStampShort field  
22 in the Security Layer Header as follows:

1           TimeStampLong = (SystemTime - (SystemTime[TimeStampShortLength-1:0] -  
2           TimeStampShort) mod  $2^{\text{TimeStampShortLength}}$ ) mod  $2^{\text{TimeStampLongLength}}$ ,

3                           Where:

4                           SystemTime is the current CDMA System Time in units time specified  
5                           by TimeStampUnit for the channel specified by the ChannelID,  
6                           TimeStampShort is a field included in the Security Protocol Header  
7                           for the channel specified by the ChannelID,  
8                           and SystemTime[n-1:0] is the  $n$  least significant bits of the  
9                           SystemTime.

10 Counter                If CounterLength is not zero and CounterExplicit is not zero, then  
11                           this field is set to the Counter field included in the Security Protocol  
12                           Header. If CounterExplicit is zero and CounterLength is not zero,  
13                           then this field shall be set to the number of Security Layer packets  
14                           (modulo  $2^{\text{CounterLength}}$ ) that have been processed since time T such  
15                           that  $T \bmod \text{TimeStampUnit} = 0$ , where T is the CDMA System Time in  
16                           units of slots. If CounterLength is zero, then this field is omitted.

17 BandClass            This field shall be set to the band class number associated with the  
18                           CDMA Channel on which the packet was received.

19 ChannelNumber       This field shall be set to the channel number associated with the  
20                           CDMA Channel on which the packet was received.

21 CounterExplicit, CounterLength, TimeStampLongLength, TimeStampShortLength, and  
22 KeyIndexLength for each channel identified by the ChannelID are configurable parameters.

### 23 2.6.2. Message Formats

24 No messages are defined for the InUse instance of this protocol.

### 25 2.6.3. Time-Counter-Based Security Protocol Header

26 The Time-Counter-Based Security Protocol Header is as specified in Table 2.6.1-3.

### 27 2.6.4. Time-Counter-Based Security Protocol Trailer

28 The Time-Counter-Based Security Protocol does not add a trailer.

### 29 2.6.5. Interface to Other Protocols

#### 30 2.6.5.1. Commands

31 This protocol does not issue any commands.

#### 32 2.6.5.2. Indications

33 This protocol does not register to receive any indications.

1 **2.7. Configuration Attributes**

2 The following complex attributes and default values are defined for this protocol. See [1] for  
 3 attribute record definition.

4 The following table specifies the complex attributes associated with the Time-Counter-  
 5 Based Security Protocol.

6 2.7.1. FTC Cryptosync Attribute  
 7

Field	Length (bits)	Default Value (decimal)
Length	8	N/A
AttributeID	8	N/A

One or more of the following record:

ValueID	8	N/A
FTCKeyIndexLength	3	0
FTCTimeStampShortLength	8	8
FTCTimeStampLongLength	8	48
FTCTimeStampUnit	4	6
FTCCounterExplicit	1	1
FTCCounterLength	8	8

8 Length Length of the complex attribute in octets. The sender shall set this  
 9 field to the length of the complex attribute excluding the Length field.

10 AttributeID The sender shall set this field to 0x00.

11 ValueID This field identifies this particular set of values for the attribute. ~~The  
 12 access network shall increment this field for each complex attribute-  
 13 value record for a particular attribute.~~

14 FTCKeyIndexLength The sender shall set this field to the value of the KeyIndexLength  
 15 which specifies the length of the KeyIndex field (in bits) in the  
 16 Security Protocol Header for the Security Layer Packets that are to be  
 17 sent on the Forward Traffic Channel.

18 FTCTimeStampShortLength  
 19 The sender shall set this field to the value of the  
 20 TimeStampShortLength which specifies the length of the  
 21 TimeStampShort Counter field (in bits) in the Security Protocol  
 22 Header for the Security Layer Packets that are to be sent on the  
 23 Forward Traffic Channel.

1	FTCTimeStampLongLength	
2		The sender shall set this field to the length of the TimeStampLong
3		field ( <a href="#">in bits</a> ) to be used in computation of the <del>cryptosync</del> <a href="#">Cryptosync</a>
4		for the Forward Traffic Channel <del>in units of bits</del> .
5	FTCTimeStampUnit	The sender shall use this field to specify the value of the
6		TimeStampUnit. This field specifies the unit of the TimeStampShort
7		field (in slots) in the Security Protocol Header for the Security Layer
8		Packets that are to be sent on the Forward Traffic Channel. The
9		value specified by TimeStampUnit shall be 2 to the power of the value
10		of this field in units of slots <sup>5</sup> .
11	FTCCounterExplicit	The sender shall set this field to 1 to request that the Counter field to
12		be included in the Security Layer Header Packets that are to be sent
13		on the Forward Traffic Channel. The sender shall set this field to 0 to
14		request that the Counter field not to be included in the Security
15		Layer Header Packets that are to be sent on the Forward Traffic
16		Channel.
17	FTCCounterLength	The sender shall set this field to the value of the CounterLength
18		which specifies the length of the Counter field (in bits) in the Security
19		Protocol Header for the Security Layer Packets that are to be sent on
20		the Forward Traffic Channel.
21	2.7.2. RTC Cryptosync Attribute	
22		

---

<sup>5</sup> For example, if the value of the FTCTimeStampUnit is 6, then the units of TimeStampShort (in the Security Protocol Header for the Security Layer Packets that are to be sent on the Forward Traffic Channel) is 64 slots.

Field	Length (bits)	Default Value (decimal)
Length	8	N/A
AttributeID	8	N/A

One or more of the following record:

ValueID	8	N/A
RTCKeyIndexLength	3	0
RTCTimeStampShortLength	8	0
RTCTimeStampLongLength	8	56
RTCTimeStampUnit	4	4
RTCCounterExplicit	1	0
RTCCounterLength	8	0

- 1 Length Length of the complex attribute in octets. The sender shall set this
- 2 field to the length of the complex attribute excluding the Length field.
  
- 3 AttributeID The sender shall set this field to 0x01.
  
- 4 ValueID This field identifies this particular set of values for the attribute. ~~The~~
- 5 ~~access network shall increment this field for each complex attribute-~~
- 6 ~~value record for a particular attribute.~~
  
- 7 RTCKeyIndexLength The sender shall set this field to the value of the KeyIndexLength
- 8 which specifies the length of the KeyIndex field (in bits) in the
- 9 Security Protocol Header for the Security Layer Packets that are to be
- 10 sent on the Reverse Traffic Channel.
  
- 11 RTCTimeStampShortLength
- 12 The sender shall set this field to the value of the
- 13 TimeStampShortLength which specifies the length of the
- 14 TimeStampShort Counter field (in bits) in the Security Protocol
- 15 Header for the Security Layer Packets that are to be sent on the
- 16 Reverse Traffic Channel.
  
- 17 RTCTimeStampLongLength
- 18 The sender shall set this field to the length of the TimeStampLong
- 19 field (in bits) to be used in computation of the ~~cryptosync~~ Cryptosync
- 20 for the Reverse Traffic Channel ~~in units of bits~~.
  
- 21 RTCTimeStampUnit The sender shall use this field to specify the value of the
- 22 TimeStampUnit. This field specifies the unit of the TimeStampShort
- 23 field (in slots) in the Security Protocol Header for the Security Layer
- 24 Packets that are to be sent on the Reverse Traffic Channel. The value

1 specified by TimeStampUnit shall be 2 to the power of the value of  
2 this field in units of slots.

3 RTCCounterExplicit The sender shall set this field to 1 to request that the Counter field to  
4 be included in the Security Layer Header Packets that are to be sent  
5 on the Reverse Traffic Channel. The sender shall set this field to 0 to  
6 request that the Counter field not to be included in the Security  
7 Layer Header Packets that are to be sent on the Reverse Traffic  
8 Channel.

9 RTCCounterLength The sender shall set this field to the value of the CounterLength  
10 which specifies the length of the Counter field (in bits) in the Security  
11 Protocol Header for the Security Layer Packets that are to be sent on  
12 the Reverse Traffic Channel.

### 13 2.7.3. CC Cryptosync Attribute

14

Field	Length (bits)	Default Value (decimal)
Length	8	N/A
AttributeID	8	N/A

One or more of the following record:

ValueID	8	N/A
CCKeyIndexLength	3	2
CCTimeStampShortLength	8	12
CCTimeStampLongLength	8	54
CCTimeStampUnit	4	6
CCCounterExplicit	1	1
CCCounterLength	8	2

15 Length Length of the complex attribute in octets. The sender shall set this  
16 field to the length of the complex attribute excluding the Length field.

17 AttributeID The sender shall set this field to 0x02.

18 ValueID This field identifies this particular set of values for the attribute. ~~The  
19 access network shall increment this field for each complex attribute  
20 value record for a particular attribute.~~

21 CCKeyIndexLength The sender shall set this field to the value of the KeyIndexLength  
22 which specifies the length of the KeyIndex field (in bits) in the

- 1 Security Protocol Header for the Security Layer Packets that are to be  
2 sent on the Control Channel.
- 3 CCTestampShortLength  
4 The sender shall set this field to the value of the  
5 TimeStampShortLength which specifies the length of the  
6 TimeStampShort Counter field (in bits) in the Security Protocol  
7 Header for the Security Layer Packets that are to be sent on the  
8 Control Channel.
- 9 CCTestampLongLength  
10 The sender shall set this field to the length of the TimeStampLong  
11 field (in bits) to be used in computation of the ~~cryptosync~~-Cryptosync  
12 for the Control Channel ~~in units of bits~~.
- 13 CCTestampUnit The sender shall use this field to specify the value of the  
14 TimeStampUnit. This field specifies the unit of the TimeStampShort  
15 field (in slots) in the Security Protocol Header for the Security Layer  
16 Packets that are to be sent on the Control Channel. The value  
17 specified by TimeStampUnit shall be 2 to the power of the value of  
18 this field in units of slots.
- 19 CCCounterExplicit The sender shall set this field to 1 to request that the Counter field to  
20 be included in the Security Layer Header Packets that are to be sent  
21 on the Control Channel. The sender shall set this field to 0 to  
22 request that the Counter field not to be included in the Security  
23 Layer Header Packets that are to be sent on the Control Channel.
- 24 CCCounterLength The sender shall set this field to the value of the CounterLength  
25 which specifies the length of the Counter field (in bits) in the Security  
26 Protocol Header for the Security Layer Packets that are to be sent on  
27 the Control Channel.

## 1 2.7.4. AC Cryptosync Attribute

2

Field	Length (bits)	Default Value (decimal)
Length	8	N/A
AttributeID	8	N/A

One or more of the following record:

ValueID	8	N/A
ACKeyIndexLength	3	2
ACTimeStampShortLength	8	10
ACTimeStampLongLength	8	52
ACTimeStampUnit	4	6
ACCounterExplicit	1	1
ACCounterLength	8	4

3 Length Length of the complex attribute in octets. The sender shall set this  
4 field to the length of the complex attribute excluding the Length field.

5 AttributeID The sender shall set this field to 0x03.

6 ValueID This field identifies this particular set of values for the attribute. ~~The  
7 access network shall increment this field for each complex attribute-  
8 value record for a particular attribute.~~

9 ACKeyIndexLength The sender shall set this field to the value of the KeyIndexLength  
10 which specifies the length of the KeyIndex field (in bits) in the  
11 Security Protocol Header for the Security Layer Packets that are to be  
12 sent on the Access Channel.

13 ~~ACCounterLength~~ ~~The sender shall set this field to the value of the CounterLength~~  
14 ~~which specifies the length of the Counter field (in bits) in the Security~~  
15 ~~Protocol Header for the Security Layer Packets that are to be sent on~~  
16 ~~the Access Channel.~~

17 ACTimeStampShortLength  
18 The sender shall set this field to the value of the  
19 TimeStampShortLength which specifies the length of the  
20 TimeStampShort ~~Counter~~ field (in bits) in the Security Protocol  
21 Header for the Security Layer Packets that are to be sent on the  
22 Access Channel.

- 1    ACTimeStampLongLength  
 2            The sender shall set this field to the length of the TimeStampLong  
 3            field (in bits) to be used in computation of the ~~cryptosync~~Cryptosync  
 4            for the Access Channel ~~in units of bits~~.
  
- 5    ACTimeStampUnit    The sender shall use this field to specify the value of the  
 6            TimeStampUnit. This field specifies the unit of the TimeStampShort  
 7            field (in slots) in the Security Protocol Header for the Security Layer  
 8            Packets that are to be sent on the Access Channel. The value  
 9            specified by TimeStampUnit shall be 2 to the power of the value of  
 10           this field in units of slots.
  
- 11   ACCounterExplicit   The sender shall set this field to 1 to request that the Counter field to  
 12           be included in the Security Layer Header Packets that are to be sent  
 13           on the Access Channel. The sender shall set this field to 0 to request  
 14           that the Counter field not to be included in the Security Layer Header  
 15           Packets that are to be sent on the Access Channel.
  
- 16   ~~ACKeyIndexLength   The sender shall set this field to the value of the KeyIndexLength  
 17           which specifies the length of the KeyIndex field (in bits) in the  
 18           Security Protocol Header for the Security Layer Packets that are to be  
 19           sent on the Access Channel.~~
  
- 20   ACCounterLength    The sender shall set this field to the value of the CounterLength  
 21           which specifies the length of the Counter field (in bits) in the Security  
 22           Protocol Header for the Security Layer Packets that are to be sent on  
 23           the Access Channel.

24   **2.8. Protocol Numeric Constants**

25

Constant	Meaning	Value
NSPType	Type field for this protocol	0x08
NSPTimeCounter	Subtype field for this protocol	0x0002

26   **2.9. Session State Information**

27    The Session State Information record (see [1]) consists of parameter records.  
 28    This protocol defines the following parameter records in addition to the configuration  
 29    attributes for this protocol.

## 1 2.9.1. KeyIndex Parameter

2 **Table 2.9.1-1. The Format of the Parameter Record for the KeyIndex Counter**  
3 **Parameter**

Field	Length (bits)
ParameterType	8
Length	8
KeyIndex	8

4 ParameterType This field shall be set to 0x01.

5 Length This field shall be set to the length of this parameter record in units  
6 of octets excluding the Length field.7 KeyIndex If the Key Exchange protocol provides the KeyIndex as one of its  
8 public data, then the sender shall set this field to the 8 LSBs of the  
9 current value of KeyIndex. Otherwise, the sender shall set this field to  
10 zero.

## 11 2.9.2. TimeStampLong Parameter

12 **Table 2.9.2-1. The Format of the Parameter Record for the TimeStampLong**  
13 **Parameter**

Field	Length (bits)
ParameterType	8
Length	8
<u>TimeStampLongLength</u>	<u>8</u>
TimeStampLong	<u>TimeStampLongLe ngth (Length - 1) x 8</u>
<u>Reserved</u>	<u>0 - 7 (as needed)</u>

14 ParameterType This field shall be set according to Table 2.9.2-2.

1

**Table 2.9.2-2. Encoding of the ParameterType Field**

Field Value	Meaning
0x02	TimeStampLong for FTC
0x04	TimeStampLong for RTC
0x06	TimeStampLong for CC
0x08	TimeStampLong for AC

2 Length

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

3

4 [TimeStampLongLength](#)

The sender shall set this field to the length of the TimeStampLong field (in bits). This field shall be set to the corresponding TimeStampLongLength field in Cryptosync attribute for the channel specified by the ParameterType field.

9 TimeStampLong

This field shall be set to the value of the TimeStampLong that has been used the last time security (i.e., authentication or encryption) has been applied on the channel specified by the ParameterType field.

10

11

12 [Reserved](#)

The number of bits in this field is equal to the number needed to make the length of this parameter record length an integer number of octets. This field shall be set to all zeros.

13

14

15 2.9.3. Counter Parameter

16 **Table 2.9.3-1. The Format of the Parameter Record for the Counter Parameter**

Field	Length (bits)
ParameterType	8
Length	8
<a href="#">CounterLength</a>	<u>8</u>
Counter	<a href="#">CounterLength</a> <del>(Length - 1) × 8</del>
<a href="#">Reserved</a>	<u>0 - 7 (as needed)</u>

17 ParameterType

This field shall be set according to Table 2.9.3-2.

1

**Table 2.9.3-2. Encoding of the ParameterType Field**

Field Value	Meaning
0x03	Counter for FTC
0x05	Counter for RTC
0x07	Counter for CC
0x09	Counter for AC

2 Length

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

3

4 CounterLength

The sender shall set this field to the length of the Counter field (in bits). This field shall be set to the corresponding CounterLength field in Cryptosync attribute for the channel specified by the ParameterType field.

5

6

7

8 Counter

This field shall be set to the value of the value of the Counter that has been used the last time security (i.e., authentication or encryption) has been applied on the channel specified by the ParameterType field.

9

10

11 Reserved

The number of bits in this field is equal to the number needed to make the length of this parameter record length an integer number of octets. This field shall be set to all zeros.

12

13

### 1 **3. AES ENCRYPTION PROTOCOL**

2 The AES Encryption Protocol uses the AES (a.k.a. Rijndael) procedures defined in [2] in  
3 order to encrypt the Connection Layer packets and decrypt the Authentication Protocol  
4 packets.

#### 5 **3.1. Primitives and Public Data**

##### 6 3.1.1. Commands

7 This protocol does not define any commands.

##### 8 3.1.2. Return Indications

9 This protocol does not return any indications.

##### 10 3.1.3. Public Data

- 11 • Subtype for this protocol

#### 12 **3.2. Protocol Data Unit**

13 The protocol data unit for this protocol is an Encryption Protocol Packet.

#### 14 **3.3. Protocol Initialization**

##### 15 3.3.1. Protocol Initialization for the InConfiguration Protocol Instance

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

- 18 • The fall-back values of the attributes for this protocol instance shall be set to the  
19 default values specified for each attribute.
- 20 • If the InUse instance of this protocol has the same protocol subtype as this  
21 InConfiguration protocol instance, then the fall-back values of the attributes defined  
22 by the InConfiguration protocol instance shall be set to the values of the  
23 corresponding attributes associated with the InUse protocol instance.
- 24 • The value for each attribute for this protocol instance shall be set to the fall-back  
25 value for that attribute.

##### 26 ~~3.3.2. Protocol Initialization for the InUse Protocol Instance~~

27 ~~Upon creation of the InUse instance of this protocol, the access terminal and the access~~  
28 ~~network shall set the value of the attributes for this protocol instance to the default values~~  
29 ~~specified for each attribute.~~

1 **3.4. Procedures and Messages for the InConfiguration Instance of the Protocol**

2 3.4.1. Procedures

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

5 3.4.2. 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  
10 InUse 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  
12 instance, 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, and
  - 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  
19 protocol instance, then the access terminal and the access network shall perform  
20 the following:
  - 21 – The InConfiguration protocol instance shall become the InUse protocol instance  
22 for the Encryption Protocol.
- 23 • All the public data not defined by this protocol shall be removed from the public  
24 data of the InUse protocol.

25 3.4.3. Message Formats

26 3.4.3.1. ConfigurationRequest

27 The ConfigurationRequest message format is as follows:

28

Field	Length (bits)
MessageID	8
TransactionID	8

Zero or more instances of the following record

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

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

1 TransactionID The sender shall increment this value for each new  
 2 ConfigurationRequest message sent.

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

4

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

5 3.4.3.2. ConfigurationResponse

6 The ConfigurationResponse message format is as follows:

7

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

8 MessageID The sender shall set this field to 0x51.

9 TransactionID The sender shall set this value to the TransactionID field of the  
 10 corresponding ConfigurationRequest message.

11 AttributeRecord An attribute record containing a single attribute value. If this  
 12 message selects a complex attribute, only the ValueID field of the  
 13 complex attribute shall be included in the message. The format of the  
 14 AttributeRecord is given in [1]. The sender shall not include more  
 15 than one attribute record with the same attribute identifier.  
 16

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

17 **3.5. Procedures and Messages for the InUse Instance of the Protocol**

18 3.5.1. Procedures

19 3.5.1.1. Constructing the Encryption Key

20 The AES Encryption Protocol shall construct the encryption keys as follows:

- 21 • If the value of the FTCEncryption attribute is equal to 0x01, then the protocol shall  
 22 construct the encryption key for the Forward Traffic Channel, FTCEncryptionKey, as  
 23 follows:

- 1           – If the Key Exchange Protocol does not define FACEncKey as public data, the  
2           access terminal shall set FTCEncryptionKey to zero of length 128 bits.
- 3           – Otherwise, the access terminal shall perform the following:
- 4           + If the length of FACEncKey is equal to 128, then FTCEncryptionKey shall be  
5           set to FACEncKey.
- 6           + Otherwise, if the length of FACEncKey is greater than 128, then  
7           FTCEncryptionKey shall be the 128 most significant bits of FACEncKey.
- 8           + Otherwise, if the length of FACEncKey is less than 128, then  
9           FTCEncryptionKey shall be the concatenation of zeros at the end (LSB) of  
10          FACEncKey, such that the length of the result is 128.
- 11          – The protocol shall perform the following:
- 12          + Call the KeyStrengthRedAlg procedure specified in [3] ~~[2]~~ with its inputs set  
13          as follows:
- 14           ◦ Set the *KeyLength* to 16.
- 15           ◦ Set the *OriginalKey* to the value of the FTCEncryptionKey.
- 16           ◦ Set the *SaltLength* to the value of the FTCSaltLength parameter.
- 17           ◦ Set the *Salt* to the value of the FTCSalt parameter.
- 18           ◦ Set the *KeyEntropy* to the value of the FTCKeyEntropy parameter.
- 19          + When the KeyStrengthRedAlg procedure returns, set the FTCEncryptionKey  
20          to *RedStrengthKey* which is the output of the KeyStrengthRedAlg procedure.
- 21          • If the value of the RTCEncryption attribute is equal to 0x01, then the protocol shall  
22          construct the encryption key for the Reverse Traffic Channel, RTCEncryptionKey, as  
23          follows:
- 24          – If the Key Exchange Protocol does not define RACEncKey as public data, the  
25          access terminal shall set RTCEncryptionKey to zero of length 128 bits.
- 26          – Otherwise, the protocol shall perform the following:
- 27          + If the length of RACEncKey is equal to 128, then RTCEncryptionKey shall be  
28          set to RACEncKey.
- 29          + Otherwise, if the length of RACEncKey is greater than 128, then  
30          RTCEncryptionKey shall be the 128 most significant bits of RACEncKey.
- 31          + Otherwise, if the length of RACEncKey is less than 128, then  
32          RTCEncryptionKey shall be the concatenation of zeros at the end (LSB) of  
33          RACEncKey, such that the length of the result is 128.
- 34          – The protocol shall perform the following:
- 35          + Call the KeyStrengthRedAlg procedure specified in [3] ~~[2]~~ with its inputs set  
36          as follows:
- 37           ◦ Set the *KeyLength* to 16.

- 1           ◦ Set the *OriginalKey* to the value of the RTCEncryptionKey.
- 2           ◦ Set the *SaltLength* to the value of the RTCSaltLength parameter.
- 3           ◦ Set the *Salt* to the value of the RTCSalt parameter.
- 4           ◦ Set the *KeyEntropy* to the value of the RTCKeyEntropy parameter.
- 5           + When the KeyStrengthRedAlg procedure returns, set the RTCEncryptionKey  
6           to *RedStrengthKey* which is the output of the KeyStrengthRedAlg procedure.
- 7   • If the value of the CCEncryption attribute is equal to 0x01, then the protocol shall  
8   construct the encryption key for the Control ~~Traffic~~-Channel, CCEncryptionKey, as  
9   follows:
- 10   – If the Key Exchange Protocol does not define FPCEncKey as public data, the  
11   protocol shall set CCEncryptionKey to zero of length 128 bits.
- 12   – Otherwise, the protocol shall perform the following:
- 13   + If the length of FPCEncKey is equal to 128, then CCEncryptionKey shall be  
14   set to FPCEncKey.
- 15   + Otherwise, if the length of FPCEncKey is greater than 128, then  
16   CCEncryptionKey shall be the 128 most significant bits of FPCEncKey.
- 17   + Otherwise, if the length of FPCEncKey is less than 128, then  
18   CCEncryptionKey shall be the concatenation of zeros at the end (LSB) of  
19   FPCEncKey, such that the length of the result is 128.
- 20   – The protocol shall perform the following:
- 21   + Call the KeyStrengthRedAlg procedure specified in [3]~~[2]~~ with its inputs set as  
22   follows:
- 23   ◦ Set the *KeyLength* to 16.
- 24   ◦ Set the *OriginalKey* to the value of the CCEncryptionKey.
- 25   ◦ Set the *SaltLength* to the value of the CCSaltLength parameter.
- 26   ◦ Set the *Salt* to the value of the CCSalt parameter.
- 27   ◦ Set the *KeyEntropy* to the value of the CCKeyEntropy parameter.
- 28   + When the KeyStrengthRedAlg procedure returns, set the CCEncryptionKey to  
29   *RedStrengthKey* which is the output of the KeyStrengthRedAlg procedure.
- 30   • If the value of the ACEncryption attribute is equal to 0x01, then the protocol shall  
31   construct the encryption key for the ~~Control~~-TrafficAccess Channel,  
32   ACEncryptionKey, as follows:
- 33   – If the Key Exchange Protocol does not define RPCEncKey as public data, the  
34   protocol shall set ACEncryptionKey to zero of length 128 bits.
- 35   – Otherwise, the protocol shall perform the following:

- 1           + If the length of RPCEncKey is equal to 128, then ACEncryptionKey shall be  
2           set to RPCEncKey.
- 3           + Otherwise, if the length of RPCEncKey is greater than 128, then  
4           ACEncryptionKey shall be the 128 most significant bits of RPCEncKey.
- 5           + Otherwise, if the length of RPCEncKey is less than 128, then  
6           ACEncryptionKey shall be the concatenation of zeros at the end (LSB) of  
7           RPCEncKey, such that the length of the result is 128.
- 8           – The protocol shall perform the following:
- 9           + Call the KeyStrengthRedAlg procedure specified in [3]~~[2]~~ with its inputs set as  
10          follows:
- 11          ◦ Set the *KeyLength* to 16.
- 12          ◦ Set the *OriginalKey* to the value of the ACEncryptionKey.
- 13          ◦ Set the *SaltLength* to the value of the ACSaltLength parameter.
- 14          ◦ Set the *Salt* to the value of the ACSalt parameter.
- 15          ◦ Set the *KeyEntropy* to the value of the ACKeyEntropy parameter.
- 16          + When the KeyStrengthRedAlg procedure returns, set the ACEncryptionKey to  
17          *RedStrengthKey* which is the output of the KeyStrengthRedAlg procedure.

### 18 3.5.1.2. Constructing the Cryptosync

19 The protocol shall construct the Cryptosync for each of the channels as follows:

- 20          • If the Security Protocol does not define Cryptosync as its public data for a channel,  
21          then the protocol shall set the Cryptosync for that channel to zero. Otherwise, this  
22          protocol shall use the value of the Cryptosync associated with the Security Layer  
23          packet given as public data by the Security Protocol.
- 24          • If the Security Protocol does not define CryptosyncLength as its public data for a  
25          channel, then the protocol shall set the CryptosyncLength field for that channel to  
26          64. Otherwise, this protocol shall use the value of the CryptosyncLength for the  
27          channel that is given as public data by the Security Protocol.

### 28 3.5.1.3. Transmit Procedures

29 The protocol shall construct the Encryption Protocol packet from the Connection Layer  
30 packet that is destined for FTC, RTC, CC or AC by performing the following for each of the  
31 channels:

- 32          • If the Encryption attribute for the channel under consideration (e.g.,  
33          FTCEncryption) is equal to 0x01, the protocol shall perform the following:
- 34          – The protocol shall call the ESP\_AES procedure specified in [2] with its inputs set  
35          as follows:
- 36                  + Set the *key* to the EncryptionKey for the channel under consideration (e.g.,  
37                  FTCEncryptionKey).

- 1           + Set *fresh* to the value of the Cryptosync for the channel under consideration.
- 2           + Set the *freshsize* to the value of the CryptosyncLength for the channel under
- 3           consideration (e.g., FTCCryptosyncLength).
- 4           + Set the *buf* to the address of the beginning of the memory space that
- 5           contains the Connection Layer packet.
- 6           + Set the *bit\_offset* to zero.
- 7           + Set the *bit\_count* to the length of the Connection Layer Packet in bits.
- 8           – After the ESP\_AES procedure is returned, the protocol shall set the Encryption
- 9           Protocol packet to the output of the ESP\_AES procedure which starts at the
- 10          memory space specified by *buf* and is of the same size as the Connection Layer
- 11          packet.
- 12          • If the Encryption attribute for the channel under consideration (e.g.,
- 13          FTCEncryption) is equal to 0x00, the protocol shall set the encryption protocol
- 14          packet to the Connection Layer packet.

#### 15 3.5.1.4. Receive Procedures

16 If the Encryption Protocol packet is received on the FTC or RTC, then the receiver shall  
 17 construct the Connection Layer packet from the Encryption Protocol packet by performing  
 18 the following ~~for each of the channels~~:

- 19          • If the Encryption attribute for the channel under consideration (e.g.,
- 20          FTCEncryption) is equal to 0x01, the protocol shall perform the following:
  - 21           – The protocol shall call the ESP\_AES procedure specified in [2] with its inputs set
  - 22           as follows:
    - 23           + Set the *key* to the EncryptionKey for the channel under consideration (e.g.,
    - 24           FTCEncryptionKey).
    - 25           + Set *fresh* to the value of the Cryptosync for the channel under consideration.
    - 26           + Set the *freshsize* to the value of the CryptosyncLength for the channel under
    - 27           consideration.
    - 28           + Set the *buf* to the address of the beginning of the memory space that
    - 29           contains the Encryption Protocol packet.
    - 30           + Set the *bit\_offset* to zero.
    - 31           + Set the *bit\_count* to the length of the Encryption Protocol Packet in bits.
  - 32           – After the ESP\_AES procedure is returned, the protocol shall set the Connection
  - 33           Layer packet to the output of the ESP\_AES procedure which starts at the
  - 34           memory space specified by *buf* and is of the same size as the Encryption
  - 35           Protocol packet.
- 36          • If the Encryption attribute for the channel under consideration (e.g.,
- 37          FTCEncryption) is equal to 0x00, the protocol shall set the Connection Layer packet
- 38          to the Encryption Protocol packet.

1 If the Encryption Protocol packet is received on the CC or AC, then the receiver shall  
2 construct the Connection Layer packet from the Encryption Protocol packet by performing  
3 the following:

- 4 • If security has been applied to the Security Layer packet<sup>6</sup> and the Encryption  
5 attribute for the channel under consideration (e.g., CCEncryption) is equal to 0x01,  
6 the protocol shall perform the following:
  - 7 – The protocol shall call the ESP\_AES procedure specified in [2] with its inputs set  
8 as follows:
    - 9 + Set the *key* to the EncryptionKey for the channel under consideration (e.g.,  
10 CCEncryptionKey).
    - 11 + Set *fresh* to the value of the Cryptosync for the channel under consideration.
    - 12 + Set the *freshsize* to the value of the CryptosyncLength for the channel under  
13 consideration.
    - 14 + Set the *buf* to the address of the beginning of the memory space that  
15 contains the Encryption Protocol packet.
    - 16 + Set the *bit\_offset* to zero.
    - 17 + Set the *bit\_count* to the length of the Encryption Protocol Packet in bits.
  - 18 – After procedure is ESP\_AES returns, the protocol shall set the Connection Layer  
19 packet to the output of the ESP\_AES procedure which starts at the memory  
20 space specified by *buf* and is of the same size as the Encryption Protocol packet.
- 21 • Otherwise, the protocol shall set the Connection Layer packet to the Encryption  
22 Protocol packet.

### 23 3.5.2. Message Formats

24 No messages are defined for the InUse instance of this protocol.

### 25 3.5.3. AES Encryption Protocol Header

26 The AES Encryption Protocol does not add a header.

### 27 3.5.4. AES Encryption Protocol Trailer

28 The AES Encryption Protocol does not add a trailer.

### 29 3.5.5. Interface to Other Protocols

#### 30 3.5.5.1. Commands

31 This protocol does not issue any commands.

---

<sup>6</sup> In the Default Control Channel MAC protocol, the SecurityLayerFormat in the MAC layer header determines whether or not security has been applied to the packet.

## 1 3.5.5.2. Indications

2 This protocol does not register to receive any indications.

3 **3.6. Configuration Attributes**

4 The configurable simple attributes for this protocol are listed in the following tables.

5 The default value for each attribute is typed in ***bold italics***.6 **Table 3.5.5-1. FTCEncryption**

<b>Attribute ID</b>	<b>Attribute</b>	<b>Values</b>	<b>Meaning</b>
0x00	FTCEncryption	<b><i>0x00</i></b>	Connection Layer packets destined for the FTC shall not be encrypted by the sender and shall not be decrypted by the receiver.
		0x01	Connection Layer packets destined for the FTC shall be encrypted by the sender and shall be decrypted by the receiver.
		0x02-0xff	Reserved

7 **Table 3.5.5-2. RTCEncryption**

<b>Attribute ID</b>	<b>Attribute</b>	<b>Values</b>	<b>Meaning</b>
0x01	RTCEncryption	<b><i>0x00</i></b>	Connection Layer packets destined for the RTC shall not be encrypted by the sender and shall not be decrypted by the receiver.
		0x01	Connection Layer packets destined for the RTC shall be encrypted by the sender and shall be decrypted by the receiver.
		0x02-0xff	Reserved

8 **Table 3.5.5-3. CCEncryption**

<b>Attribute ID</b>	<b>Attribute</b>	<b>Values</b>	<b>Meaning</b>
0x02	CCEncryption	<b><i>0x00</i></b>	Connection Layer packets destined for the CC shall not be encrypted by the sender and shall not be decrypted by the receiver.
		0x01	Connection Layer packets destined for the CC shall be encrypted by the sender and shall be decrypted by the receiver.
		0x02-0xff	Reserved

1

**Table 3.5.5-4. ACEncryption**

Attribute ID	Attribute	Values	Meaning
0x03	ACEncryption	0x00	Connection Layer packets destined for the AC shall not be encrypted by the sender and shall not be decrypted by the receiver.
		0x01	Connection Layer packets destined for the AC shall be encrypted by the sender and shall be decrypted by the receiver.
		0x02-0xff	Reserved

2 The following complex attributes are defined for reduction of the encryption key strength for  
3 each channel.

4 3.6.1. FTCReducedStrengthEncryptionKey Attribute

5

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A

One or more of the following record:

ValueID	8	N/A
FTCSaltLength	8	0
FTCSalt	FTCSaltLength × 8	N/A
FTCKeyEntropy	8	16

6 Length Length of the complex attribute in octets. The sender shall set this  
7 field to the length of the complex attribute excluding the Length field.

8 AttributeID The sender shall set this field to 0x04.

9 ValueID This field identifies this particular set of values for the attribute. ~~The  
10 sender shall increment this field for each complex attribute value  
11 record for a particular attribute.~~

12 FTCSaltLength The sender shall set this field to the length of the FTCSalt field in  
13 octets.

14 FTCSalt The sender shall set this field to the value of the *Salt* input parameter  
15 that is to be used in the KeyStrengthRedAlg procedure specified in  
16 [3]~~[2]~~ for the FTC encryption key.

17 FTCKeyEntropy The sender shall set this field to the value of the *KeyEntropy* input  
18 parameter that is to be used in the KeyStrengthRedAlg procedure

1 specified in [3]~~[2]~~ for the FTC encryption key. The valid values for  
 2 this field are 0 through 16, inclusive.

3 3.6.2. RTCReducedStrengthEncryptionKey Attribute  
 4

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A

One or more of the following record:

ValueID	8	N/A
RTCSaltLength	8	0
RTCSalt	RTCSaltLength × 8	N/A
RTCKeyEntropy	8	16

5 Length Length of the complex attribute in octets. The sender shall set this  
 6 field to the length of the complex attribute excluding the Length field.

7 AttributeID The sender shall set this field to 0x05.

8 ValueID This field identifies this particular set of values for the attribute. ~~The  
 9 sender shall increment this field for each complex attribute value  
 10 record for a particular attribute.~~

11 RTCSaltLength The sender shall set this field to the length of the RTCSalt field in  
 12 octets.

13 RTCSalt The sender shall set this field to the value of the *Salt* input parameter  
 14 that is to be used in the KeyStrengthRedAlg procedure specified in  
 15 [3]~~[2]~~ for the RTC encryption key.

16 RTCKeyEntropy The sender shall set this field to the value of the *KeyEntropy* input  
 17 parameter that is to be used in the KeyStrengthRedAlg procedure  
 18 specified in [3]~~[2]~~ for the RTC encryption key. The valid values for  
 19 this field are 0 through 16, inclusive.

20 3.6.3. CCReducedStrengthEncryptionKey Attribute  
 21

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A

One or more of the following record:

ValueID	8	N/A
CCSaltLength	8	0
CCSalt	CCSaltLength × 8	N/A
CCKeyEntropy	8	16

- 1 Length Length of the complex attribute in octets. The sender shall set this  
2 field to the length of the complex attribute excluding the Length field.
- 3 AttributeID The sender shall set this field to 0x06.
- 4 ValueID This field identifies this particular set of values for the attribute. ~~The  
5 sender shall increment this field for each complex attribute value  
6 record for a particular attribute.~~
- 7 CCSaltLength The sender shall set this field to the length of the CCSalt field in  
8 octets.
- 9 CCSalt The sender shall set this field to the value of the *Salt* input parameter  
10 that is to be used in the KeyStrengthRedAlg procedure specified in  
11 [3]~~[2]~~ for the CC encryption key.
- 12 CCKeyEntropy The sender shall set this field to the value of the *KeyEntropy* input  
13 parameter that is to be used in the KeyStrengthRedAlg procedure  
14 specified in [3]~~[2]~~ for the CC encryption key. The valid values for this  
15 field are 0 through 16, inclusive.

#### 16 3.6.4. ACReducedStrengthEncryptionKey Attribute

17

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A

One or more of the following record:

ValueID	8	N/A
ACSaltLength	8	0
ACSalt	ACSaltLength × 8	N/A
ACKeyEntropy	8	16

- 1 Length Length of the complex attribute in octets. The sender shall set this  
2 field to the length of the complex attribute excluding the Length field.
- 3 AttributeID The sender shall set this field to 0x07.
- 4 ValueID This field identifies this particular set of values for the attribute. ~~The  
5 sender shall increment this field for each complex attribute value  
6 record for a particular attribute.~~
- 7 ACSaltLength The sender shall set this field to the length of the ACSalt field in  
8 octets.
- 9 ACSalt The sender shall set this field to the value of the *Salt* input parameter  
10 that is to be used in the KeyStrengthRedAlg procedure specified in  
11 [3][2] for the AC encryption key.
- 12 ACKKeyEntropy The sender shall set this field to the value of the *KeyEntropy* input  
13 parameter that is to be used in the KeyStrengthRedAlg procedure  
14 specified in [3][2] for the AC encryption key. The valid values for this  
15 field are 0 through 16, inclusive.

16 **3.7. Protocol Numeric Constants**

Constant	Meaning	Value
N <sub>EPT</sub> type	Type field for this protocol	0x07
N <sub>EPAES</sub>	Subtype field for this protocol	0x0001

18 **3.8. Session State Information**

- 19 The Session State Information record (see [1]) consists of parameter records.
- 20 The parameter records for this protocol consist of only the configuration attributes of this  
21 protocol.

- 1 |
- 2 | [No text.](#)
- 3 |