情 報 保 護 學 會 誌
第17巻 第2號, 2007. 4

社團法人 電子情報通信學會
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

情報通信 システムセキュリティ研究會
Information and Communication System Security
ICSS2006-14 (2007-02)

Invited Speech at ICSS 2007

Generation of Session, Authentication, and Encryption Keys for CDMA2000 1x EV-DO Air Interface Standard

Man Young Rhee*

Abstract

The air interface supports a security layer which provides the key exchange protocol, authentication protocol, and encryption protocol. The authentication is performed on the encryption protocol packet. The authentication protocol header or trailer may contain the digital signature that is used to authenticate a portion of the authentication protocol packet that is authenticated. The encryption protocol may add a trailer to hide the actual length of the plaintext of padding to be used by the encryption algorithm. The encryption protocol header may contain variables such as the initialization vector (IV) to be used by the encryption protocol. It is our aim to firstly compute the session key created from the D H key exchange algorithm, and thereof the authenticating key and the encryption key being generated from the session key.

1. Introduction

The air interface between the access terminal (AT) and the access network (AN) contains a security layer in its reference protocol layering, as shown in Figure 1. The security layer provides the key exchange protocol, authentication protocol, and encryption protocol. This security layer can be used for encryption and authentication of an excess terminal traffic transported by the control channel, the access channel, the forward traffic channel (FTC) and reverse traffic channel (RTC). The security protocol provides the following functions between the AT and the AN:

- To exchange security keys for authentication and encryption
- To provide the procedures for authenticating traffic and encryption traffic
- To provide public variables (i.e, cryptosync, time-stamps, etc) needed by the authentication and encryption protocol

* Endowed Chair Professor, Kyung Hee University(myrhee@tsp.snu.ac.kr)

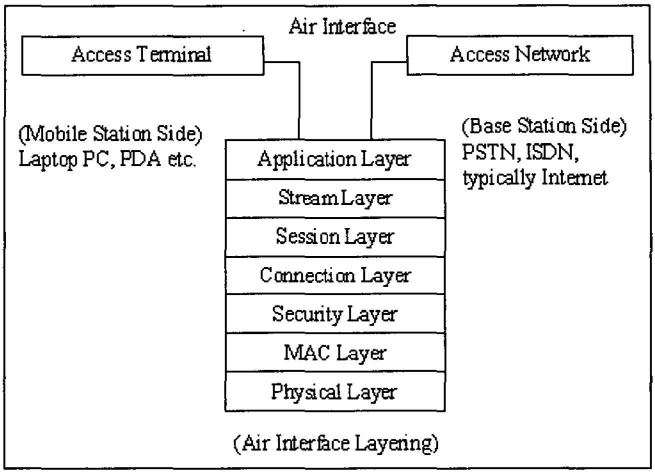

Figure 1. Reference model architecture

Figure 2 illustrates the relationship between a connection layer packet, a security layer packet and a MAC layer payload. The authentication is performed on the encryption protocol packet and the portion of security layer packet that may be encrypted and authenticated. The data unit for the security protocol is security layer packet. Each security layer packet consists of an authentication protocol packet, and security layer header or trailer. Specifically, the encryption protocol may add a trailer to hide the actual length of the plaintext or padding to be used by the encryption algorithm. The encryption protocol header may contain variables such as the initialization vector (IV) and cryptosync to be used by the encryption protocol. The authentication protocol header or trailer may contain the digital signature that is used to authenticate the portion of the authentication protocol packet that is authenticated.

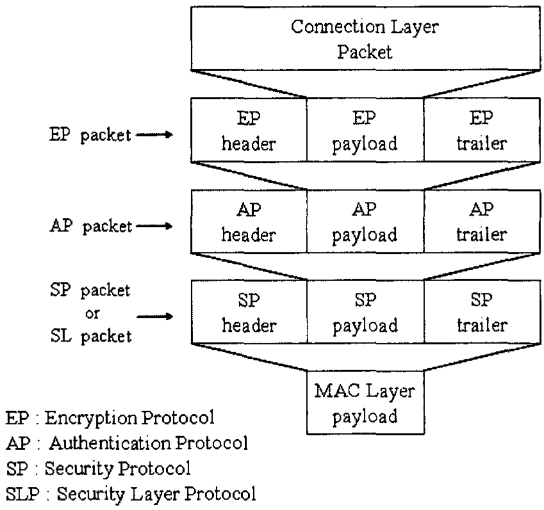

Figure 2. Security layer protocols between the connection layer and MAC layer

The security protocol header or trailer may contain variables needed by the authentication and encryption protocols. However, the security layer headers and trailers may not be present if the session configuration established the default security layer or if the security protocol does not require a header or trailer. The fields added by the MAC layer will indicate presence or absence of the security layer headers and trailers.

The security protocol constructs the authentication protocol packet using the security layer packet received from the MAC layer and forwards the packet to the authentication protocol. The protocol for the InUse instance should set the authentication protocol packet to the security layer packet received from the MAC layer and should forward the packet to the authentication protocol. The generic security protocol on the transmission side provides a cryptosync that may be used by the negotiated authentication protocol and encryption protocol, whereas on the receiving side, this protocol computes the cryptosync using the information provided in the security protocol header and makes the cryptosync publicly available.

The computation of the Session Key (SKey) being created from the D-H key exchange protocol, authentication key, and encryption key generated from the session key will be presented in the following sections.

2. Session Key Generation

The D-H key exchange algorithm provides a method for session key (SKey) exchanges that can be applied for use on the KeyRequest and KeyResponse messages for exchanging public session keys, and the ANKeyComplete and ATKeyComplete messages for indicating that the secret session keys have been calculated.

The AT and the AN perform the following key exchange procedure during session configuration.

Access Network (AN) Requirements

The AN sends the KeyRequest message to initiate the session key exchange. The AT will send the KeyResponse message in response to the KeyRequest message. The AN chooses a random number ANRand between $1 < \text{ANRand} \leq p-2$ and sets the ANPubKey field of the KeyRequest message as follows:

$$\text{ANPubKey} = g^{\text{ANRand}} \pmod p$$

where g, p are KeyLength dependent protocol constants and KeyLength is specified during session configuration of the D-H key exchange protocol.

The random number ANRand has the following properties:

- The generated ANRand should have a uniform distribution over its range and should be statistically uncorrelated in formulating different KeyRequest messages.
- Formation of each KeyRequest message should not be derivable from the random number used previously.
- The number used in formulating KeyRequest message sent by different access networks should be statistically uncorrelated.

After receiving a KeyResponse message with a TransactionID field (8bits) that matches the TransectionID field (8bits) of the associated KeyRequest message, the AN should perform the following:

- The AN computes SKey (session key) such that
  SKey = ATPubKeyANRand (mod p)

- The AN constructs the message bits which consist of the computed SKey, TransactionID (8 bits), a 16-bit pseudorandom (Nonce), and TimeStampLong (64 bits). The 64-bit TimeStamLong is set based on the current 64-bit representation of the CDMA System Time in units of 80ms.

- The AN pads the message bits and computes the 160-bit message digest.

- The AN sends an ANKeyComplete message with the KeySignature field of the messages set to the message digest computed in the previous step and the TimeStampShort field of the message set to the 16 least significant bits of the CDMA System Time used in the previous step. The AN then starts the AT Signature Computation Timer with a timeout value of the AN Key Signature Computation Timer.

The AN should declare failure and stop performing the rest of the key exchange procedure if either the AT key computation and the AT key signature computation timers are expired, or the AN receives an ATKeyComplete message with Result field set to '0'.

Access Terminal (AT) Requirements

Upon receiving the KeyRequest message, the AT should perform as follows:

- Choose a random number ATRand between $1 < \text{ATRand} \leq p\text{-}2$ and sets the ATPubKey field of the KeyResponse message as follows:
  $\text{ATPubKey} = g^{\text{ATRand}} \pmod{p}$
  Where g, p are KeyLength dependent protocol constants for the D-H key exchange protocol.

- The AT sends a KeyResponse message with the A TPubKey field.

- The AT computes the session key (SKey) as follows:
  SKey = ANPubKeyATRand(mod p)

After receiving an ANKeyComplete message with a TransactionID field (8bits) that matches the TransectionID field (8bits) of the associated KeyRequest message, the AT should perform the following:

- The AT computes the 64-bit variable TimeStampLong as follows
  $TimeStampLong = (SystemTime - (SystemTime[15:0] - TimeStampShort) \bmod 2^{16})$
  where SystemTime is the current CDMA System Time (80ms), SystemTime[15:0] is the least significant bits

of the System Time, and TimeStampShort is the 16-bit field received in the ANKeyComplete message.

- The AT pads the message bits and computes the 160-bit message digest.

- If the message digest matches the KeySignature field of ANKeyComplete message, the AT sends an ATKeyComplete message with the result field set to '1'.

Otherwise, the AT declares failure and sends an ATKeyComplete message with the result field set to '0'.

3. Applications and Practical Examples

The session key exchange procedures during session configuration between the AT and the AN are summarized in the followings.

Table 1 D-H Key Exchange Algorithm for SKey Computation

AN		AT
Choose ANRand, 1 ≤ ANRand ≤ p-2		Choose ATRand, 1 ≤ ATRand ≤ p-2
Compute the AN public key:		Compute the AT public key:
$ANPubKey \equiv g^{ANRand} \pmod{p}$		$ATPubKey \equiv g^{ATRand} \pmod{p}$
Send ANPubKey to the AT	→	
	←	Send ATPubKey to the AN
Finally compute the session key:		Compute Session Key:
$SKey \equiv ATPubKey^{ANRand} \pmod{p}$ $\equiv g^{(ATRand)(ANRand)} \pmod{p}$		$SKey \equiv ANPubKey^{ATRand} \pmod{p}$ $\equiv g^{(ANRand)(ATRand)} \pmod{p}$
If the SKey computed at AN and AT is equal, the session key is established.		

where,
ANRand: Access network random number
ATRand: Access terminal random number
ANPubKey: Access network public Key
ATPubKey: Access terminal public Key
SKey: Session key
g: A primitive element of a prime modulo p
p: a prime number

Here, g and p are specified during session configurations of the D-H Key Exchange Protocol. g and p are called cryptosync.

Example 1

A small example will be presented for the reader to easily understand how to compute the session key (Skey).

Choose protocol constants g=6 and p=11 for the D-H key exchange protocol.
Pick ATRand = 3, $1 \leq$ ATRand ≤ 9. Pick ANRand = 7, $1 \leq$ ANRand ≤ 9
Compute first the public keys of the AT and AN, respectively :

$ATPubKey \equiv g^{ATRand}(mod\ p),$ $\equiv 6^3 \pmod{11} = 7$

$ATPubKey \equiv g^{ANRand}(mod\ p),$ $\equiv 6^7 \pmod{11} = 8$

AT sends *ATPubKey* to AN ⇄ AN sends *ANPubKey* to AT

Finally, AT computes the SKey :

$SKey \equiv ANPubKey^{ATRand}(mod\ p),$
$\equiv 8^3 \pmod{11} = 6$

AN computes the SKey :

$SKey \equiv ATPubKey^{ANRand}(mod\ p),$
$\equiv 7^7 \pmod{11} = 6$

Thus, the SKey (session Key) between AT and AN is computed.

The SKey computation by means of Elliptic Curve (EC) cryptography over either Z_p or $GF(2^m)$ can be done using the EC D-H key exchange algorithm as shown in Table 2 and Table 3, respectively.

Table 2 SKey Computation using EC Diffie-Hellman Key Exchange Protocol over the Prime Field Zp

AT		AN
Choose a generator $G=(x, y)$ on the EC curve and two random numbers of *ATRand* and *ANRand*.		
Compute $ATPubKey \equiv (ATRand)\cdot G$ AT sends his publicKey to AN	⇄	Compute $ANPubKey \equiv (ANRand)\cdot G$ AN also sends his publicKey to AT
Then, compute the SKey as follows : $SKey \equiv (ATRand)(ANPubKey)$ $\equiv (ATRand)(ANRand)\cdot G$		$SKey \equiv (ANRand)(ATPubKey)$ $\equiv (ANRand)(ATRand)\cdot G$
Thus, the session key is computed.		

Example 2

Suppose G(5,2) is a generator point on EC y2+ xy = x3+ x + 6
for a = 1 and b = 6. Choose a prime module p = 11.

When choosing $ATRand = 2$ *and* $ANRand = 3$, $ATPubKey = (ATRand)G$ *and* $ANPubKey = (ANRand)G$ can be computed as shown below.

Computation of the public key at AT:
$ATPubKey = 2G = 2(5,2) = (5,2) + (5,2) = (x_3, y_3)$

This is the case in which two points are doubling.

$$\beta = \frac{3x_1^2+1}{2y_1}(\mathrm{mod}\ p) = \frac{3\times 25+1}{2\times 2}(\mathrm{mod}11) = \frac{76}{4} = 19(\mathrm{mod}11) = 8$$

$$x_3 = \beta^2 - 2x_1 = 64 - 2\times 5 = 54(\mathrm{mod}11) = 10$$

$$y_3 = \beta(x_1 - x_3) - y_1 = 8(5-10) - 2 = -42(\mathrm{mod}11) = 2$$

Thus, *ATPubKey* = (10,2)

Computation of the public key at AN:

ANPubKey = *3G* = *(5,2)* + *(5,2)* + *(5,2)* = *(x_3, y_3)*

This is the case at where scalar multiplication of a point on EC is simply repeated addition of that point. Since 2G = (10,2), ANPubKey = (10,2) + (5,2) = (x_3, y_3)

$$\alpha = \frac{y_2 - y_1}{x_2 - x_1} = \frac{2-2}{5-10} = 0$$

$$x_3 = \alpha^3 - x_1 - x_2 = -10 - 5 = -15(\mathrm{mod}11) = 7$$

$$y_3 = \alpha(x_1 - x_3) - y_1 = -2(\mathrm{mod}11) = 9$$

Hence, *ANPubKey* = (7, 9)

Next, consider the computation of SKeys at AT and AN, respectively.

ATSKey = *(ATRand)(ANPubKey)* = *2(7, 9)* = *(7, 9)* + *(7, 9)*

$$\beta = \frac{3\times 49+1}{2\times 9} = \frac{74}{9} = \frac{8}{9}(\mathrm{mod}11) = 8\times 9^{-1}(\mathrm{mod}11) = 8\times 5(\mathrm{mod}11) = 7$$

$$x_3 = \beta^2 - 2x_1 = 49 - 2\times 7 = 35(\mathrm{mod}11) = 2$$

$$y_3 = \beta(x_1 - x_3) - y_1 = 7(7-2) - 9 = 26(\mathrm{mod}11) = 4$$

Thus, *ATSkey* = (2, 4)

ANSKey = *(ANRand)*(*ATPubKey*) = 3(10, 2)
= (10, 2) + (10, 2) + (10, 2)

The addition of first two terms becomes:

Since β=1, it results in x_3=3 and y_3=5

Hence, *ANSKey* = (3, 5) + (10, 2)

$$\alpha = \frac{2-5}{10-3} = -3\times 7^{-1} = -3\times 8 = -24(\mathrm{mod}\ 11) = 9$$

$$x_3 = \alpha^2 - x_1 - x_2 = 81 - 3 - 10 = 68(\mathrm{mod}\ 11) = 2$$

$$y_3 = \alpha(x_1 - x_3) - y_1 = 9(3-2) - 5 = 4$$

Thus, ANSKey = (2, 4)

Since computed results at AT and AN are identical, the session key is established.

Table 3 The SKey Computation using the EC Diffie-Hellman Key Exchange Algorithm over the Binary Extension Field GF(2m)

Select the base point (a generator) G on the EC curve $y^2+xy = x^3+ax^2+b$. Let n denote the order of GF(2^m).	
AT	**AN**
Choose $ATRand$, $1 \leq ATRand \leq n$	Choose $ANRand$, $1 \leq ANRand \leq n$
Compute $ATPubKey = (ATRand)G$	Compute $ANPubKey = (ANRand)G$
AT sends $ATPubKey$ to AN. ⇄	AN sends $ANPubKey$ to AT.
Computation of the SKey (session key)	
$ATSKey = (ATRand)(ANPubKey)$ $= (ATRand)(ANRand)G$	$ANSKey = (ANRand)(ATPubKey)$ $= (ANRand)(ATRand)G$
Thus the session key between AT and AN can be computed.	

Example 3

Consider GF(2^4) whose primitive polynomial is $p(x)=x^4+x+1$ of degree 4. If α denotes a root of $p(x)$, then the field elements α^i, $1 \leq \alpha^i \leq 15$, of GF(2^4) can be generated by using $\alpha^4 = \alpha + 1$.

Suppose the EC equation over GF(2^4) is $y^2+xy = x^3+a^4x^2+1$ *by* $a = a^4$ and $b=1$. Let the base point (a generator) be G = (α^5, α^3).

Computation at AT (Access Terminal):

Pick ATRand=3 from the range of $1 \leq$ ATRand ≤ 15.

Compute ATPubKey = (ATRand)G = 3(α^5, α^3) = (α^5, α^3) + (α^5, α^3) + (α^5, α^3)

Let's find the solution of the first two doubling points.

(α^5, α^3) + (α^5, α^3) = (x_3, y_3) = (1, α^{13})

$$x_3 = x_1^2 + \frac{b}{x_1^2} = (\alpha^5)^2 + \frac{1}{(\alpha^5)^2} = \alpha^{10} + \alpha^5 = 1$$

$$y_3 = x_1^2 + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3 = (\alpha^5)^2 + \left(\alpha^5 + \frac{\alpha^3}{\alpha^5}\right) + 1$$

$$= a^{10} + a^5 + a^{13} + 1 = a^{13}$$

Thus, ATPubKey = (1, a^{13}) + (a^5, a^3) = (x_3, y_3)

This is the case for adding two distinct points on the elliptic curve.

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2} = \frac{\alpha^{13} + \alpha^3}{1 + \alpha^5} = \frac{1 + \alpha^2}{\alpha^{10}} = \alpha^{13}$$

$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a = \alpha^{26} + \alpha^{13} + 1 + \alpha^5 + \alpha^4 = \alpha^2 + 1 + \alpha = \alpha^{10}$$

$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1 = \alpha^{13}(1 + \alpha^{10}) + \alpha^{10} + \alpha^{13} = \alpha^{10} + \alpha^8 = \alpha$$

Finally, ATPubKey = (a^{10}, a)

AT sends ATPubKey = (a^{10}, a) to AN.

Computation at AN (Access Network):

Let us pick ANRand = 2 from the range of $1 \leq$ ANRand ≤ 15.

Compute ANPubKey = (ANRand)G = $2(a^5, a^3) = (a^5, a^3) + (a^5, a^3) = (1, a^{13})$

AN sends ANPubKey = $(1, a^{13})$ to AT.

Lastly, the SKey (session key) is computed as follows:

AT computes the ATSKey = (ATRand)(ANPubKey)

= $3(1, a^{13}) = (1, a^{13}) + (1, a^{13}) + (1, a^{13})$

For the first two doubling points, we have

$(1, a^{13}) + (1, a^{13}) = (x_3, y_3) = (0, 1)$

$$x_3 = x_1^2 + \frac{b}{x_1^2} = 1 + 1 = 0$$

$$y_3 = x_1^2 + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3 = 1 + 0 + 0 = 1$$

ATSKey = $(0, 1) + (1, a^{13}) = (x_3, y_3)$

Applying the formula for the addition of two distinct points on EC, we have

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2} = \frac{1 + \alpha^{13}}{0 + 1} = \alpha^2 + \alpha^3 = \alpha^6$$

$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a = (\alpha^6)^2 + \alpha^6 + 0 + 1 + \alpha^4 = \alpha^{12} + \alpha^6 + 1 + \alpha^4 = 1$$

$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1 = \alpha^6(0 + 1) + 1 + 1 = \alpha^6$$

The Session Key computed at AT is

ATSKey = $(1, a^6)$

AN computes the ANSKey = (ANRand)(ATPubKey)

= $2(a^{10},\ a) = (a^{10},\ a) + (a^{10},\ a)$

$$x_3 = x_1^2 + \frac{b}{x_1^2} = (\alpha^{10})^2 + \frac{1}{(\alpha^{10})^2} = \alpha^{20} + \alpha^{-20} = \alpha^5 + \alpha^{10} = 1$$

$$y_3 = x_1^2 + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3 = (\alpha^{10})^2 + \left(\alpha^{10} + \frac{\alpha}{\alpha^{10}}\right) + 1 = \alpha^5 + \alpha^{10} + \alpha^6 + 1 = \alpha^6$$

Thus the Session Key computed at AN is

ANSKey = $(1,\ a^6)$

Since ATSKey = ANSKey = $(1,\ a^6)$, it is proved that the session key established between AT and AN.

4. Authentication Key and Encryption Key Generation

The key used for authentication and encryption is generated from the session key (SKey), using the procedures specified in the following. SKey contains eight sub-fields (K_0, K_1, ⋯, K_7) that are of equal length (KeyLength/8). The AT and AN construct their message bits using the computed SKey, the 8-bit TransactionID, a 16-bit pseudo-random value (Nonce), and a 64-bit TimeStampLong (based on the current 64-bit representation of the CDMA System Time in units of 80ms). The AN and AT should pad the message bits and compute 160-bit message digests for each of the eight subkeys of SKey. The AN and AT then set the FTCAuthKey, RTCAuthKey, FTCEncKey, RTCEncKey, FCCAuthKey, RACAuthKey, FCCEncKey, and RACEncKey to the message digests for the corresponding sub-keys as shown in Table 4.

where FTC = Forward Traffic Channel, FCC = Forward Control Channel

RTC = Reverse Traffic Channel, RAC = Reverse Access Channel

Table 4 Message bits for Generation of Authentication and Encryption Keys

	MSB		LSB
Message bits for generation of four AuthKeys and four EncKeys	K_i $(0 \le i \le 7)$	Nonce $(0 \le i \le 7)$	TimeStampLong $(0 \le i \le 7)$
	KeyLength/8	16 bits	64 bits

AuthKeys and EncKeys corresponding to message digests computed from eight sub-fields within the SKey are shown in the following example.

In Figure 3, the schematic diagram of SHA-1 algorithm is shown.

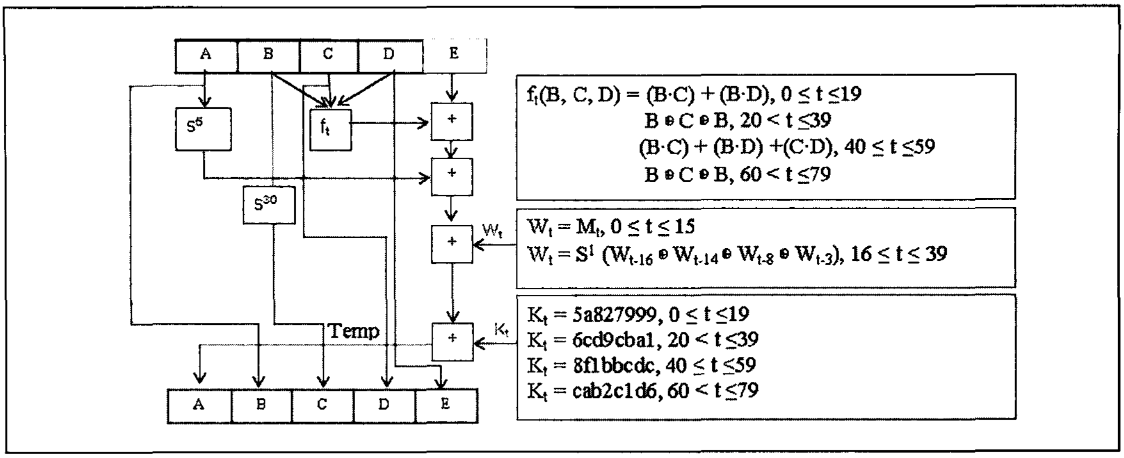

Figure 3. Schematic diagram of SHA-1 algorithm

A, B, C, D, and E are initialized by the following initial constants as shown in Figure 3:

A = H0 = 67452301
B = H1 = efcdab89
C = H2 = 98badcfe
D = H3 = 10325476
E = H4 = c3d2e1f0

Temp = $S^5(A) + F_t(B,C,D) + E + W_t + K_t$
E = D; D = C; C = S^{30} (B); B = A; A = Temp

t: round number, $0 \le t \le 79$
S^i : circular left shift by i bits
K_t : four distinct additive constants over $0 \le t \le 79$
W_t : a 32-bit word derived from the current 512-bit input block
⊞ : addition module 2^{32}

Example 4

Consider the 128-bit SKey which contains eight sub-fields of equal length, i.e., $K_0 = K_1 = K_2 = K_3 = K_4 = K_5 = K_6 = K_7 = 16$ bits.

The AT and the AN pad the message bits to produce a 512-bit padded message and compute the 160-bit digests by means of SHA-1 hash function that can be used for computation of authentication and encryption keys.

Suppose SKey (session key) is given by:

SKey =	20a7	4f12	30ae	c5ff	52ad	621b	8601	a901
	K_0	K_1	K_2	K_3	K_4	K_5	K_6	K_7

Firstly, FTCAuthKey enables to compute as shown below:

FTCAuthkey = K_0 || TransactionID || Nonce || TimeStampLong
= 16 bits || 8 bits || 16 bits || 64 bits

where,
K_0 = 0x 20a7 (16 bits)
TransactionID = 0x 10 (8 bits)
Nonce = 0x f263 (16 bits)
TimeStampLong = 0x a801 4cff 5106 de22 (64bits)

The padded message bits, as shown below, should be used for computation of 160-bit message digest as follows (use big-endian form):

First append a '1' followed by padding of 312 '0'

0x	20a710f 2	63a8014c	ff5106de	22800000
	00000000	00000000	00000000	00000000
	00000000	00000000	00000000	00000000
	00000000	00000000	00000000	00000068

Length Field (last two words: 00000000 00000068)

Let A, B, C, D, and E are initialized by the following initial constants:

A = H0 = 67452301
B = H1 = efcdab89
C = H2 = 98badcfe
D = H3 = 10325476
E = H4 = c3d2e1f0

Through 80 continuous operations, according to SHA-1 algorithm, the hex values of A, B, C, D and E are computed as shown below:

Steps	A	B	C	D	E
0	c05ba9a5	67452301	7bf36ae2	98badcfe	10325476
1	d5ce0311	c05ba9a5	59d148c0	7bf36ae2	98badcfe
2	28400a71	d5ce0311	7016ea69	59d148c0	7bf36ae2
3	590e7d61	28400a71	757380c4	7016ea69	59d148c0
4	467a4ecc	590e7d61	4a10029c	757380c4	7016ea69
5	0654be0e	467a4ecc	56439f58	4a10029c	757380c4

Steps	A	B	C	D	E
6	e8cfca75	654be0e	119e93b3	56439f58	4a10029c
7	ea35e44	e8cfca75	81952f83	119e93b3	56439f58
8	16c6fcf5	ea35e44	7a33f29d	81952f83	119e93b3
9	d0381f75	16c6fcf5	03a8d791	7a33f29d	81952f83
10	4dcd6e6f	d0381f75	45b1bf3d	3a8d791	7a33f29d
11	d21519d4	4dcd6e6f	740e07dd	45b1bf3d	03a8d791
12	e50b2321	d21519d4	d3735b9b	740e07dd	45b1bf3d
13	37b3bcab	e50b2321	34854675	D3735b9b	740e07dd
14	fb797197	37b3bcab	7942c8c8	34854675	d3735b9b
15	ce2ad377	fb797197	cdecef2a	7942c8c8	34854675
16	ddb944aa	ce2ad377	fede5c65	cdecef2a	7942c8c8
17	dd0c56c1	ddb944aa	f38ab4dd	fede5c65	cdecef2a
18	bc6a6b38	dd0c56c1	b76e512a	F38ab4dd	fede5c65
19	d91585e4	bc6a6b38	774315b0	B76e512a	f38ab4dd
20	05fc91ec	d91585e4	2f1a9ace	774315b0	b76e512a
21	646a9fc0	5fc91ec	36456179	2f1a9ace	774315b0
22	05c5151e	646a9fc0	017f247b	36456179	2f1a9ace
23	b3280fa3	05c5151e	191aa7f0	017f247b	36456179
24	ad124529	b3280fa3	81714547	191aa7f0	017f247b
25	2da70a62	ad124529	ecca03e8	81714547	191aa7f0
26	ed42f31b	2da70a62	6b44914a	ecca03e8	81714547
27	c1e39894	ed42f31b	8b69c298	6b44914a	ecca03e8
28	7c496883	c1e39894	fb50bcc6	8b69c298	6b44914a
29	3a269e3c	7c496883	3078e625	fb50bcc6	8b69c298
30	611294c1	3a269e3c	df125a20	3078e625	fb50bcc6
31	1d6efe82	611294c1	0e89a78f	df125a20	3078e625
32	bff8d7c8	1d6efe82	5844a530	0e89a78f	df125a20
33	144cde03	bff8d7c8	875bbfa0	5844a530	0e89a78f
34	c4ae5995	144cde03	2ffe35f2	875bbfa0	5844a530
35	f34e286	c4ae5995	c5133780	2ffe35f2	875bbfa0
36	c3fcfe2a	f34e286	712b9665	C5133780	2ffe35f2
37	f50a3003	c3fcfe2a	83cd38a1	712b9665	c5133780
38	e47f696b	f50a3003	b0ff3f8a	83cd38a1	712b9665
39	1b1881b	e47f696b	fd428c00	b0ff3f8a	83cd38a1
40	a3f67ca6	1b1881b	f91fda5a	fd428c00	b0ff3f8a
41	2280ce98	a3f67ca6	c06c6206	F91fda5a	fd428c00
42	1a853ddf	2280ce98	a8fd9f29	C06c6206	f91fda5a
43	c29dbea	1a853ddf	8a033a6	A8fd9f29	c06c6206
44	b9e5504c	c29dbea	c6a14f77	8a033a6	a8fd9f29
45	34bee947	b9e5504c	830a76fa	C6a14f77	8a033a6

Steps	A	B	C	D	E
46	7cd54db5	34bee947	2e795413	830a76fa	c6a14f77
47	c69f0ffc	7cd54db5	cd2fba51	2e795413	830a76fa
48	ea7e9a7f	c69f0ffc	5f35536d	cd2fba51	2e795413
49	38577dfb	ea7e9a7f	31a7c3ff	5f35536d	cd2fba51
50	111acd7b	38577dfb	fa9fa69f	31a7c3ff	5f35536d
51	20d1a54b	111acd7b	ce15df7e	fa9fa69f	31a7c3ff
52	774c48ad	20d1a54b	c446b35e	ce15df7e	fa9fa69f
53	d394b159	774c48ad	c8346952	C446b35e	ce15df7e
54	e2fd3511	d394b159	5dd3122b	C8346952	c446b35e
55	263848cc	e2fd3511	74e52c56	5dd3122b	c8346952
56	0ef68ed	263848cc	78bf4d44	74e52c56	5dd3122b
57	5acd4cec	0ef68ed	98e1233	78bf4d44	74e52c56
58	d5f5f742	5acd4cec	403bda3b	98e1233	78bf4d44
59	438477c0	d5f5f742	16b3533b	403bda3b	98e1233
60	e1cb3109	438477c0	b57d7dd0	16b3533b	403bda3b
61	f5d1c599	e1cb3109	10e11df0	B57d7dd0	16b3533b
62	293a0d2e	f5d1c599	7872cc42	10e11df0	b57d7dd0
63	2c9a7086	293a0d2e	7d747166	7872cc42	10e11df0
64	a5c7c557	2c9a7086	8a4e834b	7d747166	7872cc42
65	16a0f2bf	a5c7c557	8b269c21	8a4e834b	7d747166
66	97190c09	16a0f2bf	e971f155	8b269c21	8a4e834b
67	94774b92	97190c09	c5a83caf	E971f155	8b269c21
68	f4a8c8d9	94774b92	65c64302	C5a83caf	e971f155
69	88281df4	f4a8c8d9	a51dd2e4	65c64302	c5a83caf
70	e45fe640	88281df4	7d2a3236	A51dd2e4	65c64302
71	a1e6b49	e45fe640	220a077d	7d2a3236	a51dd2e4
72	769795f2	a1e6b49	3917f990	220a077d	7d2a3236
73	c084d7d1	769795f2	42879ad2	3917f990	220a077d
74	c6054c69	c084d7d1	9da5e57c	42879ad2	3917f990
75	ef4439c1	c6054c69	702135f4	9da5e57c	42879ad2
76	43488a63	ef4439c1	7181531a	702135f4	9da5e57c
77	d2bef450	43488a63	7bd10e70	7181531a	702135f4
78	8117f963	d2bef450	d0d22298	7bd10e70	7181531a
79	e9b48aac	8117f963	34afbd14	D0d22298	7bd10e70

Finally,

$H_0 = H_0 + A = 50f9adad$

$H_1 = H_1 + B = 70e5a4ec$

$H_2 = H_2 + C = cd6a9a12$

$H_3 = H_3 + D = e104770e$

$H_4 = H_4 + E = 3fa3f060$

Now the 160-bit message digest is calculated by concatenating the H0, H1, H2, H3, and H4:
50f9adad70e5a4eccd6a9a12e104770e3fa3f060

Thus, the computed FTCAuthkey by using K_0 = 20a7 is
50f9adad70e5a4eccd6a9a12e104770e3fa3f060.

Similarly, the following keys are computed by utilizing Ki for i= 1, 2, ⋯, 7, respectively:
RTCAuthKey = 8691443d86a13cda16bc75c1e8332b077f287262
FTCEncKey = 2de7b52c5f95473a94a12ad5f44a467c6d65153c
RTCEncKey = ba15cb193662d15c8d7bf18ef71291fe5c783968
FCCAuthKey = bf08eb2aeee452349d418e9e6f7a3ce51cdbe02b
RACAuthKey = 344cb44161dfeccdf8be13b1f1f0955544f1cadd
FCCEncKey = 8e019fab7e5a85668eefa3adbd9cfc64bcee2798
RACEncKey = abb0380ab055c0c47761d14126fad8dbcc5bb01f

5. Conlusion

The air interface between the access terminal (AN) and the access network (AN) contains a security layer which provides the key exchange protocol, authentication protocol, and encryption protocol. It is clearly shown by the examples how to compute the authentication keys and encryption keys for authentication and encryption of the traffics transported by FTC, RTC, FCC, and RAC. These keys are generated from the session key created from the Diffie-Hellman key exchange algorithm. It will be helpful for the beginner to study wireless mobile communication security.

References

[1] IMT 2000 3GPP2 C.S0002-A v6.0, February 2002
[2] IMT 2000 3GPP2 C.S0002-B v1.0, April 2002
[3] IMT 2000 3GPP2 C.S0005-C v0.02, May 2002
[4] IMT 2000 3GPP2 C.S0024-A v1.0, March 2004
[5] FIPS Pub 180-2, Secure Hash Standard, August 2002