Securing Communications over ATM Networks

Xun Yi, Chee Kheong Siew, Yuan Miao

Information Communication Institute of Singapore
School of Electrical and Electronic Engineering
Nanyang Technological University
Nanyang Ave., Singapore 639798
E-mail: {exyi,ecksiew, eymiao}@ntu.edu.sg

Abstract

In February 1999, the ATM forum international consortium approved the first version of its security specifications, aiming to protect communications over Asynchronous Transfer Mode (ATM) networks by offering confidentiality, integrity and authentication services for various security levels. This paper shows that this specification can be further improved if Digital Signature Algorithm (DSA) is widely supported by ATM end systems. In this paper, we come up with a new proposal for securing communications over ATM networks, which initially relies on DSA. In comparison with the ATM Security Specifications Version 1.0, our proposal is more secure and efficient under the assumption that DSA is available.

Key words: *B-ISDN, ATM, ATM security, security services.*

1. Introduction

To suit the needs of multimedia services in terms of *Quality of Service* (*QoS*), *ITU-T* has defined the *Broadband Integrated Services Digital Networks* (*B-ISDN*) and adopted the *Asynchronous Transfer Model* (ATM) as the technology to implement *B-ISDN*.

ATM delivers important advantages over existing *LAN* and *WAN* technologies, including the promise of scalable bandwidths at unprecedented price and performance points and *Quality of Service* (*QoS*) guarantees.

ATM is cell switching and connection-oriented technology which is usually represented as a three-plane models. The user plane is responsible for user data exchange over ATM networks. The control plane monitors signaling information. The management plane maintains the network operations. When an application needs to send user data over the ATM network, three lower layers are involved in ATM cells construction. *ATM Adaptation Layer* (*AAL*) receives user data from the upper layers application and segments these data into 48-byte blocks which are then sent to the ATM layer. The ATM layer appends to each block a 5-byte header thus obtaining 53-byte ATM cells. The 5-byte header encompasses routing information useful for the network to transport cells to the appropriate destination. ATM cells are then sent by physical layer over the transmission medium.

As other networks, ATM networks suffer a lot of threats [1][2][3][4][5]. Typical ones are eavesdropping, masquerade (or spoofing), service denial, *Virtual Channel* (*VC*) stealing and traffic analysis etc.. The *VC* stealing and traffic analysis happen only in ATM networks. To build an ATM security system, the first thing we should do is to identify the requirements of securing communications over ATM networks. This issue has been discussed widely in ATM Forum [6][7][8][9] and literatures [1][2][3][4].

The *ATM Security Model* can be illustrated in figure 1, in which SA_i ($i=1,2,3,4$) are security agents which initiate, establish, provide, discontinue, or terminate any of security services, such as access control, authentication, confidentiality and data integrity. The initiator (or calling user) *A* and the responder (or called user) *B* are end systems or endpoints of the ATM networks. In this security model, an intruder is assumed to be able to wire-tap ATM networks and switches, record all traffic passing through ATM network and switches, replay old messages and inject his own information into the communication stream.

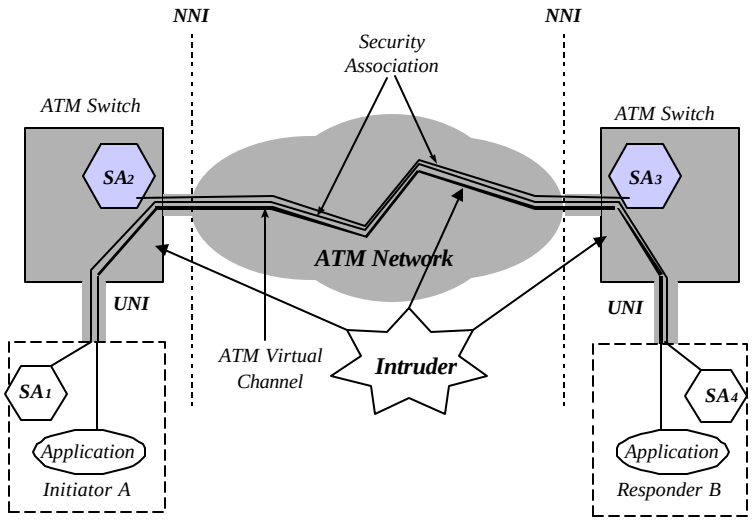

Figure 1. The ATM Security Model

To secure ATM communications, it is necessary to introduce the following security services:

- Signaling protection by offering the authentication and integrity services.
- Security parameters negotiation.
- Data protection by ensuring the confidentiality and integrity of data.

Since 1995, the ATM Forum consortium and other working groups have been working on introducing security services into ATM networks [2][3][4][5][9][10][11][12][13][14]. This does not consist in defining new security mechanisms, but in reusing existing security mechanisms such as encryption, digital signatures and etc.. In 1997, the ATM Forum has published the "*Phase I* ATM *Security Specification*" [9]. This is the first step in providing clear procedure for implementing security services in ATM networks. In February 1999, the ATM forum international consortium approved the first version of its security specifications [13] that is available on the Forum's web site: www.atmforum.com. This specification tries to provide confidentiality, integrity and authentication service over ATM networks for various security levels.

In our opinion, the *ATM Security Specifications Version 1.0* can be further improved if *Digital Signature Algorithm* (*DSA*) is supported by ATM end systems. *DSA* is a variant of the ElGamal scheme [15]. With *DSA* available, signaling messages for connection(s) (such as *SETUP* and *CONNECT*) can be rapidly authenticated before negotiating security parameters. It can thus quickly establish point-to-point or point-to-multipoint connections. It is reasonable

to assume that *DSA* is desirable because *DSA* has become a *U.S. Federal Information Processing Standard* (*FIPS 186*) called the *Digital Signature Standard* (*DSS*) [16] and is the first digital signature scheme recognized by most of governments. In addition, *Certification Authorities* (*CAs*) have to come to a standard digital signature algorithm to issue certificates.

In this paper, we come up with a new proposal for securing communications over ATM networks, which initially relies on *DSA*. It is carried out in three stages: (1) rapidly authenticating signaling messages for connection(s) (such as *SETUP* and *CONNECT*) by inserting security information elements into them; (2) securely negotiating security parameters in the user plane; (3) effectively applying negotiated security parameters in the user data exchange. This new proposal has three particular features: (1) key agreement protocol is carried out during the authentication of signaling messages for connection(s); (2) security options during negotiation are protected by the ElGamal encryption scheme based on *Discrete Logarithm* (*DL*) Problem; (3) session keys are updated synchronously on basis of the number of encrypted 48-bit blocks. In comparison with the ATM *Security Specifications Version 1.0*, our proposal is more secure and efficient under the assumption that *DSA* are available.

The following sections are arranged as follows: Section 2 introduces the ATM *Security Specifications Version 1.0*; Section 3 presents our proposal for securing communication over ATM networks; Section 4 compares the ATM *Security Specifications Version 1.0* with our proposal. Conclusion is drawn in the last section.

2. ATM Security Specifications Version 1.0

In order to negotiate parameters for the security services and to directly support the entity authentication services, two-way and three-way security message exchange (*SME*) protocols are adopted in *ATM Security Specifications Version 1.0*. The two-way *SME* protocol may be used for establishing point-to-point or point-to-multipoint connections and in particular for connections that do not require negotiation of security parameters. It is implemented with *UNI4.0* signaling. The three-way *SME* protocol may be used for establishing point-to-point connections and in particular for connections that require negotiation of security options. It is implemented in the *In-Band Security Message Exchange Protocol*. Both the two-way *SME* protocol for authentication of signaling messages for connection(s) and the three-way *SME* protocol for negotiation of security parameters are briefly introduced in this section.

The following table shows the symbols and abbreviations used throughout the discussion of this paper:

X	Entity *X*.
K_X	K_X represents the public key of *X* when it's used for encryption. K_X represents the private key of *X* when it's used for digital signature.
$Enc_{K_X}(text)$	Encryption of *text* under *X*'s key.
$Sig_{K_X}(Hash(text))$	*X*'s digital signature computed over the hash of *text* under *X*'s key where K_X is the private key of *X*.
Hash(text)	One-way hash of *text*, where *Hash* is a strong one-way hash function such as *Secure Hash Algorithm* (*SHA-1*).

R_X	Random number (nonce) generated by *X*.
T_X	Time-variant timestamp generated by *X*.
{.}	Optional token.
SecOpt	*SecOpt* token indicates to the responder what security services are to be provided for the connection.
SecNeg_	Initiator and responder use $SecNeg_a$ and $SecNeg_b$ to negotiate the security services, options, and parameters for the connection.
ConfPar_	When the key exchange support service option is invoked, $ConfPar_a$ and $ConfPar_b$ are used to securely carry one side's keys from the side to another side.
Cert_	$Cert_a$ carries the initiator's certificate or CRL chain while $Cert_b$ carries the responder's certificate or CRL chain.

2.1. Authentication of Signaling Messages for Connection(s)

In the *ATM Security Specification Version 1.0*, authentication of signaling message for point-to-point connection is achieved through the two-way *SME* protocol as shown in figure 2.

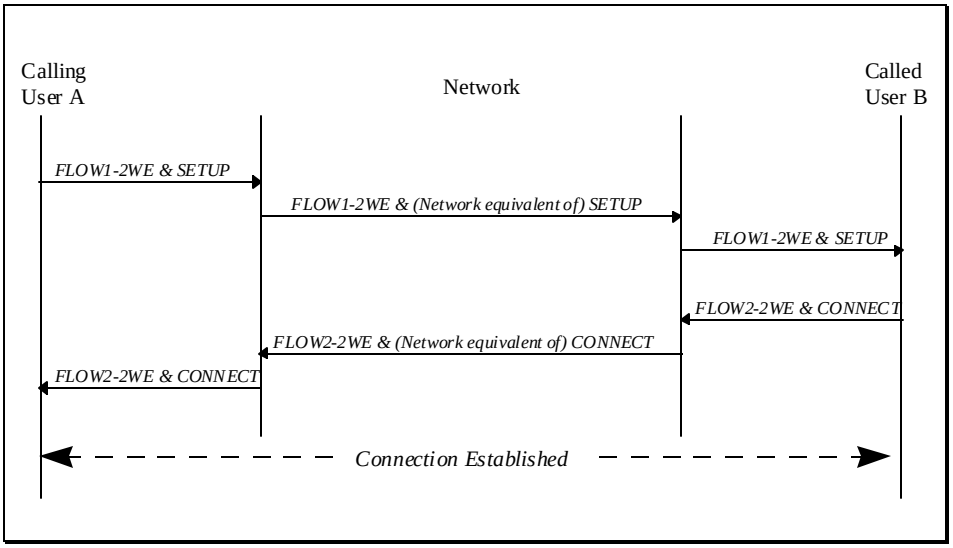

Figure 2. Authentication of Signaling Messages for Point-to-Point Connection

At the *UNI* interface, *FLOW1-2WE* is carried in the *SETUP* message and *FLOW2-2WE* is carried in the *CONNECT* message. When the calling user (or initiator) *A* wants to establish a point-to-point connection with the called user (or responder) *B*, the authentication procedure involves the following steps:

Step 1. *A* sends *FLOW1-2WE* to *B*.

FLOW1-2WE: A→B

$$A, B, SecOpt, \{Cert_a\}, \{T_a, R_a, \{Enc_{k_b}(ConfPar_a)\}, Sig_{k_a}(Hash(A, B, T_a, R_a, SecOpt, \{ConfPar_a\}))\}$$

Step 2. When *B* receives *FLOW1-2WE*, it carries out the following actions:

- Checks that *B* itself is the intended recipient.
- Extracts *SecOpt* and interprets it.
- Verifies the signature, and thus the integrity of *FLOW1-2WE*.
- Checks that the timestamp is fresh and that the flow is not a replay or out-of-order.
- Extracts the nonce R_a for its reply.
- Extracts $ConfPar_a$ if present and interprets it.
- Extracts $Cert_a$ if present and verifies its validity.

Step 3. *B* sends *FLOW2-2WE* to *A*.

FLOW2-2WE: B→A

$$\{A, B, R_a, \{Cert_b\}, \{Enc_{k_a}(ConfPar_b)\}, Sig_{k_b}(Hash(A, B, R_a, \{ConfPar_b\}))\}$$

Step 4. When *A* receives *FLOW2-2WE*, it carries out the following actions:

- Checks that *A* itself is the intended recipient.
- Verifies the signature, and thus the integrity of *FLOW2-2WE*.
- Checks that the received R_a in *FLOW2-2WE* is identical to the one which sent in *FLOW1-2WE*.
- Extracts $ConfPar_b$ if present and interprets it.
- Extracts $Cert_b$ if present and verifies its validity.

For point-to-multipoint connections, the authentication of signaling messages for the first party connection is the same as that for point-to-point connection. When setting up subsequent party, at the *UNI* interface on the calling side, *FLOW-2WE* is carried in the *ADD PARTY* message and *FLOW2-2WE* is carried in the *ADD PARTY ACK* message while at the *UNI* interface on the called side, *FLOW1-2WE* is carried in the *SETUP* message and *FLOW2-2WE* is carried in the *CONNECT* message.

2.2. Security Parameters Negotiation

In *ATM Security Specifications Version 1.0*, security parameters are negotiated through the three-way *SME* protocol which involves the following steps:

Step 1. *A* sends *FLOW1-3WE* to *B*.

FLOW1-3WE: *A→B*

$$A, \{B\}, R_a, SecNeg_a, \{Cert_a\}$$

Step 2. When *B* receives *FLOW1-3WE,* it carries out the following actions:

- Checks that *B* itself is the intended recipient, when *B* is included.
- Extracts $SecNeg_a$ and interprets it for its reply.
- Extracts the nonce R_a for its reply.
- Extracts $Cert_a$ if present and verifies its validity.

Step 3. *B* sends *FLOW2-3WE* to *A*.

FLOW2-3WE: *B*→*A*

$$A, B, SecNeg_b, \{Cert_b\}, \{R_a, R_b, \{Enc_{k_a}(ConfPar_b)\}, Sig_{k_b}(Hash(A, B, R_a, R_b, SecNeg_a, SecNeg_b, \{ConfPar_b\}))\}$$

Step 4. When *A* receives *FLOW2-3WE,* it carries out the following actions:

- Checks that *A* itself is the intended recipient.
- Extracts $SecNeg_b$ and interprets it.
- Verifies the signature, and thus the integrity of both *FLOW1-3WE* and *FLOW2-3WE*.
- Checks that the received R_a in *FLOW2-3WE* is identical to the one that sent in *FLOW1-3WE*.
- Extracts the nonce R_b for its reply.
- Extracts $ConfPar_b$ if present and interprets it.
- Extracts $Cert_b$ if present and verifies its validity.

Step 5. *A* sends *FLOW3-3WE* to *B*.

FLOW3-3WE: *A*→*B*

$$\{A, B, R_b, \{Enc_{k_b}(ConfPar_a)\}, Sig_{k_a}(Hash(A, B, R_b, \{ConfPar_a\}))\}$$

Step 6. When *B* receives *FLOW3-3WE,* it carries out the following actions:

- Checks that *B* itself is the intended recipient.
- Verifies the signature, and thus the integrity of *FLOW3-3WE*.
- Checks that the received R_b in *FLOW3-3WE* is identical to the one that sent in *FLOW2-3WE*.
- Extracts $ConfPar_a$ if present and interprets it.

2.3. Existing Problems

In the *ATM Security Specification Version 1.0*, for connection setup with the two-way *SME* protocol, when key exchange is needed, $ConfPar_a$ is included in *FLOW1-2WE* and encrypted with the called user *B*'s public key. However, the call user *A* may not know *B*'s public key when sending out the first signaling message *FLOW1-2WE*. The public key may be obtained by retrieving *B*'s public key certificate from a public key directory or by exchanging public key certificates directly during security parameters negotiation. But it will increase the number of communications and influence *Quality of Services* (*QoS*) in rapidly establishing connections.

In addition, during security parameters negotiation, security options are transmitted over ATM networks in the clear form. An intruder can know the final security options for the connection between the calling user and the called user by monitoring the traffic over ATM networks. On basis of the information, the intruder can adopt corresponding efficient cryptanalysis techniques to attack this cryptosystem. For example, if the intruder knows that the calling users and the called users are using *Data Encryption Standard* (*DES*) with 56-bit key to encrypt data, he can try to use the efficient differential cryptanalysis technique to attack the encryption system.

Finally, in *FLOW1-3WE* of three-way *SME* protocol, the calling user *A*'s signature is not required. Upon receiving *FLOW1-3WE*, the called user *B* cannot authenticate the message although *A*'s certificate is included in the message. Let us imagine the attack from a hacker. The hacker forges a lot of *FLOW1-3WE* messages with various certificates obtained over ATM networks and then sends them to a called user *B*'s server. It will result in *B*'s server becoming slow or even shutting down because the server is forced to be busy in making decisions about various security options and producing various signatures.

3. New Proposal for Securing Communications over ATM Networks

In order to overcome those problems recognized in the above section, we present a new proposal for securing communications over ATM networks in this section. Under the assumption that *DSA* is available, this new proposal is carried out in three stages: (1) rapidly authenticating signaling messages (such as *SETUP* and *CONNECT*) by inserting security information elements into them; (2) securely negotiating security parameters in the user plane; (3) effectively applying negotiated security parameters on the user data exchange. The three stages are described respectively as follows.

3.1. Authentication of Signaling Message for Connections with Key Agreement

ATM is a connection-oriented technique. A connection, which is called *Virtual Circuit* (*VC*) in ATM, is managed by a set of signal. *VC* is established by *SETUP* signals and can be disconnected by *RELEASE* or *DROP PARTY* signals. If an intruder sends *RELEASE* or *DROP PARTY* signal to any intermediate switch on the way of a *VC*, the *VC* will be disconnected. By sending these signals frequently, the intruder can greatly disturb the communication between one user to another, therefore will disable the *Quality of Service* (*QoS*) in ATM networks. Combining this technique with other tricks eavesdropping, the intruder can even completely block one user from another. In view of it, authentication is necessary for establishing or releasing each connection.

ATM network is a high speed network. It is very important to reduce the latency of the authentication protocol run. The length of such protocol run will cause an impact on the call setup performance, which is a key quality of service indicator that a network service provider can provide to customer. Therefore, rapidly authenticating signaling message is desirable for security services in ATM networks.

On establishing point-to-point connection in ATM networks, the new proposal authenticates *SETUP* and *CONNECT* signaling messages by inserting security information elements in these signaling messages in the similar way shown in figure 2. It involves the following steps:

First of all, we assume that $Cert_a$ of an entity *A* is issued by a *Certification Authorities* (*CA*) which uses *DSA* with parameters (p, q, g) to issue certificates to users. Usually, the three parameters (p, q, g) are included in each certificate so that *CA*'s signatures can be verified properly. Then

Step 1. A chooses a random integer x from *1* to *q-1* and calculates $g^x \bmod p$. Since each end system in ATM networks is supposed to support *DSA*, multiplication and exponentiation operations over *Galois fields GF(p)* (a finite field with p elements) can be computed. The calling user *A* send *FLOW1-2WE** to the called user *B*.

*FLOW1-2WE**: $A \rightarrow B$

$$A, B, Cert_a, T_a, R_a, g^x \bmod p, \ Sig_{k_a}(Hash(A, B, T_a, R_a, g^x \bmod p))$$

Step 2. When *B* receives *FLOW1-2WE**, it carries out the following actions:

- Checks that *B* itself is the intended recipient.
- Extracts $Cert_a$ and verifies its validity.
- Checks that the timestamp is fresh and that the flow is not a replay or out-of-order.
- Extracts the nonce R_a for its reply.
- Extracts $g^x \bmod p$, and chooses a random integer y from *1* to *q-1*, and calculates $g^y \bmod p$ and $K=(g^x)^y \bmod p$.
- Verifies the signature, and thus the integrity of *FLOW1-2WE**.

Step 3. *B* sends *FLOW2-2WE** to *A*.

*FLOW2-2WE**: $B \rightarrow A$

$$B, A, Cert_b, R_a, g^y \bmod p, Sig_{k_b}(Hash(B, A, R_a, g^x \bmod p, g^y \bmod p))$$

Step 4. When *A* receives *FLOW2-2WE**, it carries out the following actions:

- Checks that *A* itself is the intended recipient.
- Extracts $Cert_b$ and verifies its validity.

- Checks that the received R_a in *FLOW2-2WE** is identical to the one which sent in *FLOW1-2WE**.
- Extracts $g^y \bmod p$ and calculates $K=(g^y)^x \bmod p$.
- Verifies the signature, and thus the integrity of *FLOW2-2WE**.

Through the above two-way message exchange, the calling user *A* and the called user *B* can mutually authenticate. Besides authentication, they share a secret key $K=(g^y)^x \bmod p$ in the end. In fact, this way for key exchange just follows *Diffie-Hellman* (*DH*) *key agreement protocol* [17].

The procedure of authentication of signaling messages for point-to-point connection with key agreement can be illustrated in figure 3.

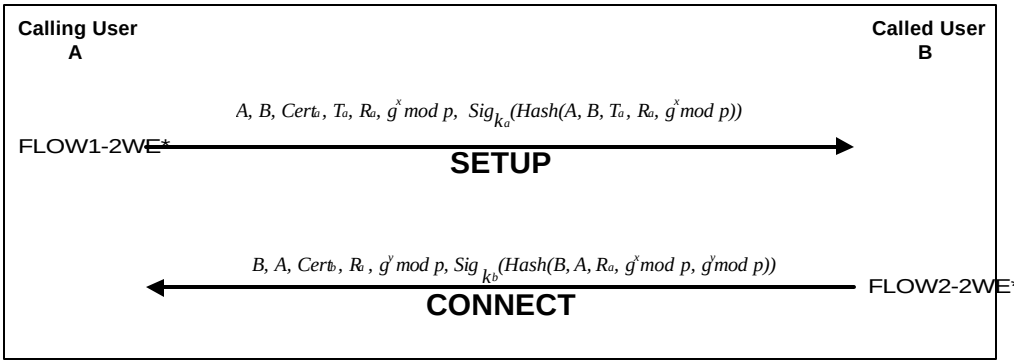

Figure 3. Authentication of Signaling Messages for Point-to-Point Connection with Key Agreement

The procedure of authentications of signaling messages for point-to-multipoint connections with key agreement is almost the same as that for point-to-point connection. It is briefly described as follows:

Suppose the calling user *A* want to establish connections with the called users $B_1, B_2, \ldots, B_n$. *A* sends the same *FLOW1-2WE** to $B_1, B_2, \ldots, B_n$ respectively.

*FLOW1-2WE**: $A \rightarrow B_1, B_2, \ldots, B_n$

$$A, B_1,\ldots,B_n, Cert_a, T_a, R_a, g^x \bmod p, Sig_{k_a}(Hash(A, B_1,\ldots,B_n, T_a, R_a, g^x \bmod p))$$

After authenticating *FLOW1-2WE**, B_i chooses a random integer $y(i)$ from *1* to $q-1$, and calculates $g^{y(i)} \bmod p$ and $K=(g^x)^{y(i)} \bmod p$, and then sends *FLOW2-2WE(i)** to *A*.

*FLOW2-2WE(i)**: $B_i \rightarrow A$

$$B_i, A, Cert_{b(i)}, R_a, g^{y(i)} \bmod p, Sig_{k_{b(i)}}(Hash(B_i, A, R_a, g^x \bmod p, g^{y(i)} \bmod p))$$

After the two-way message exchange, the calling user *A* and the called user B_i ($i=1,2,\ldots,n$) can mutually authenticate and share a secret key $K=(g^x)^{y(i)}=(g^{y(i)})^x \bmod p$.

3.2. Security Parameters Negotiation

It is clear that ATM flows require protection. However, different countries have different rules and regulations about security issues. As well, ATM connections are expected to be of various sensitive levels, it is necessary that the security services used to protect data are negotiated. Security parameters may be negotiated through signaling message, data channel or *OAM* cells.

Our proposal performs security parameters negotiation right after a connection has been set up between the calling user *A* and the called user *B*. The negotiation is carried out through two-way in-band message exchange on the user plane. At that time, although they share a secret key *K*, the agreement about which symmetric encryption algorithm will be used to protect this connection have not yet been reached between the user A and the user B. In addition, security options cannot be encrypted by *DSA*. In order to protect security options against eavesdropping, *ElGamal encryption scheme* [15] based on *Discrete Logarithm* (*DL*) problem is applied as follows.

According to the *ATM Security Specification Version 1.0*, the bit length of $SecNeg_a$ together with the initial random number R_a is 296. For *DSA* with parameters (p, q, g), p is a prime with bit length from 512 to 1024. At the end system of the user *A*, the security option $SecNeg_a$ is firstly concatenated with the initial random number R_a and a new random number R_a*. The total bit length of $(SecNeg_a||R_a||R_a^*)$ (where "||" means the concatenation of two digital blocks) has to be just one bit less than the bit length of p. For example, the total bit length of $(SecNeg_a||R_a||R_a^*)$ should be 1023 if the bit length of p is 1024. Then $(SecNeg_a||R_a||R_a^*)$ is encrypted by choosing a random integer r for *1* to *q-1* and computing

$$a_a = g^r \bmod p, \tag{1}$$

$$g_a = (SecNeg_a||R_a||R_a^*) \cdot (K_b)^r \bmod p, \tag{2}$$

where (p, q, g) and K_b (the public key of the user *B*) are determined according to $Cert_b$. The ciphertext of $(SecNeg_a||R_a||R_a^*)$ is (a_a, g_a).

Upon receiving (a_a, g_a), *B* decrypts the ciphertext by computing

$$S = (a_a)^{k_b} = (g^r)^{k_b} = (g^{k_b})^r = (K_b)^r \bmod p, \tag{3}$$

$$SecNeg_a||R_a||R_a^* = g_a \cdot S^{-1} \bmod p, \tag{4}$$

where k_b is the private key of the user *B*.

The above encryption and decryption processes only involve multiplication and exponentiation operations over *GF(p)* which are actually included in *DSA*.

On basis of *ElGamal encryption scheme*, security parameters negotiation can be carried out in the following steps:

Step 1. *A* sends *FLOW1-2WNE* to *B*.

FLOW1-2WNE: $A \rightarrow B$

$$A, B, (a_a, g_a), Sig_{k_a}(Hash(A, B, SecNeg_a||R_a||R_a^*, a_a, g_a))$$

Notice that (a_b, g_b) is computed according to formulae (1) and (2).

Step 2. When *B* receives *FLOW1-2WNE*, it carries out the following actions:

- Checking that *B* itself is the intended recipient, when *B* is included.
- Decrypting (a_a, g_a) into $SecNeg_a||R_a||R_a^*$ according to formulae (3) and (4).
- Checking that R_a in *FLOW1-2WNE* is identical to the one which sent in *FLOW1-2WE*.
- Verifying the signature and integrity of *FLOW1-2WNE*.

Step 3. *B* replies *FLOW2-2WNE* to *A*.

FLOW2-2WNE: $B \rightarrow A$

$$B, A, (a_b, g_b), Sig_{k_b}(Hash(A, B, SecNeg_b||R_a||R_b^*, a_b, g_b))$$

Notice that (a_b, g_b) is computed in the same way as formulae (1) and (2).

Step 4. When *A* receives *FLOW2-2WNE*, it carries out the following actions:

- Checking that *A* itself is the intended recipient.
- Decrypting (a_b, g_b) into $SecNeg_b||R_a||R_b^*$ in the same way as formulae (3) and (4).
- Checking that R_a in *FLOW2-2WNE* is identical to the one which sent in *FLOW1-2WE*.
- Verifying the signature and integrity of *FLOW2-2WNE*.

In the above way, user *A* and user *B* can securely reach their security parameters negotiation agreement $SecNeg_b$. The procedure of security parameter negotiation is illustrated in figure 4.

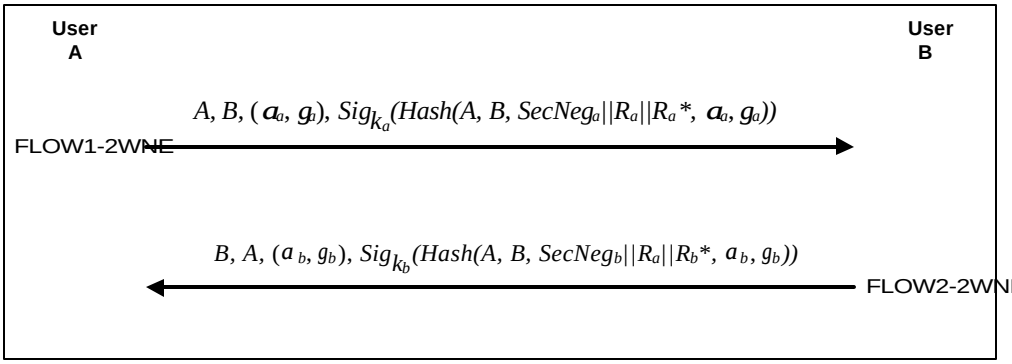

Figure 4. Security Parameters Negotiation

After the security parameters negotiation, both *A* and *B* know what encryption algorithm, digital signature algorithm, hash algorithm, key exchange algorithm, session key update algorithm and etc. will be used for this connection.

3.3. Exchange of User Data

Confidentiality of user exchange data is provided via encryption. In ATM networks, encryption can be carried out in three ways: (1) encryption at the higher layer; (2) encryption in the ATM layer; (3) encryption in the AAL layer. In the *ATM Security Specifications Version 1.0*, encryption takes place in the ATM layer as shown in figure 5.

In addition, the *ATM Security Specification Version 1.0* refers to three levels of key:

- Top-level key – asymmetric key used to authenticate and initialize the first session key and master key securely.
- Master key – symmetric key used to encrypt session keys when updating session keys during connection.
- Session key – symmetric key used to encrypt user data.

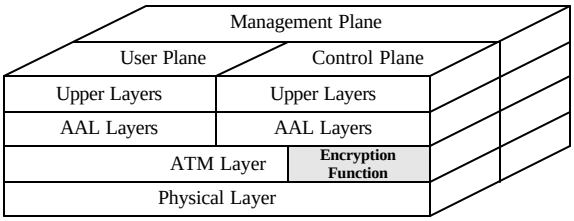

Figure 5. The Placement of Encryption Function

Our proposal follows the placement of the encryption function and the key hierarchy of the *ATM Security Specification Version 1.0*. However, the generation and update of the master key and session keys in our proposal are different from those in the *ATM Security Specification Version 1.0*.

After the connection is established between the calling user *A* and the called user *B*, the common number $K=(g^x)^y=(g^y)^x \bmod p$ is secretly shared by *A* and *B* according to *Diffie-Hellman key agreement protocol*. After the security parameters are negotiated, two more common numbers R_a* and R_b* are secretly shared by *A* and *B*. The three numbers *K*, R_a* and R_b* are only known by *A* and *B*.

In our proposal, *K* works as the master key. The first session key can be generated with the hash algorithm agreed by *A* and *B* during security parameters negotiation. If both *A* and *B* agree to use *Secure Hash Algorithm* (*SHA-1*) [18] and the bit length *L* of the secret key for the agreed symmetric encryption algorithm is less than 160, the first session key can be generated by both *A* and *B* in the following way:

- Concatenating a one-octet (8-bit) counter i, the master key K, the source's distinguished name A, the destination's distinguished name B, security options $SecNeg_a$ and $SecNeg_b$, three random numbers R_a, R_a* and R_b* together as follows:

  $$R = (i||K||i||A||i||B||i||SecNeg_a||i||SecNeg_b||i||R_a||i||R_a^*||i||R_b^*)$$

  where i=1.
- Appending *KeyFill* to the above result by using the pad with the length technique defined for *SHA-1*, i.e, $R||KeyFill$.
- Computing the following value k_1 by using *SHA-1*.

  $$k_1=Hash(R||KeyFill)$$
- Determining the first session key k by cutting the leftmost L bits from k_1 because the bit length of *SHA-1* output is 160 and $L \leq 160$.

If $L > 160$, the first session key k can be determined by computing of k_1, k_2, … and then cutting the leftmost L bits from the concatenations of $k_1||k_2||\ldots$.

Since *DSA* contains *Secure Hash Algorithm* (*SHA-1*), *SHA-1* is also supposed to be supported by each end system in ATM networks. Therefore, each end system can adopt the above method to generate the first session key.

Due to the high volume data transmitted over ATM networks, the lift time of any session key become very short. Thus, there is a requirement for mechanism to change the session key rapidly during the lifetime of a call. In order to suit the requirement, the session key is updated synchronously on basis of the length of encrypted message in our proposal. Hence, it is not necessary to deliver the new session key from the source to the destination(s).

Similar to generation of the first session key, the new session key k* (supposed to have bit length L less than 160) is generated by the master key K, the current key k and other shared secret knowledge in the following way:

- Concatenating a one-octet (8-bit) counter i, the current session key k, the master key K, the source's distinguished name A, the destination's distinguished name B, security options $SecNeg_a$ and $SecNeg_b$, three random numbers R_a, R_a* and R_b* together as follows:

  $$R = (i||k||i||K||i||A||i||B||i||\ SecNeg_a||i||SecNeg_b||i||R_a||i||R_a^*||i||R_b^*)$$

  where i=1.
- Appending *KeyFill* to the above result by using the pad with length technique defined for *SHA-1*, i.e, $R||KeyFill$.
- Computing the following value k_1 by using *SHA-1*.

  $$k_1=Hash(R||KeyFill)$$
- Determining the new session key k* by cutting the leftmost L bits from k_1.

If $L > 160$, the new session key k* can be determined by computing of k_1, k_2, … and then cutting the leftmost L bits from the concatenations of $k_1||k_2||\ldots$.

In fact, session keys can be computed by both *A* and *B* in advance and stored in memory for future use.

In our proposal, the key update synchronization points are determined by the length of encrypted message. During security parameters negotiation, *A* and *B* may reach an agreement which regulates that a session key is updated just after *N* blocks (each block has 48 bits) of user data are encrypted under this session key. For example, suppose *A* encrypt N_1 blocks under a session key and transmit them to *B* while *B* encrypt N_2 blocks under the same session key and transmit them to *A*. Once $N_1+N_2=N$, both *A* and *B* update session key. If $N=10240$, the amount of exchanged message encrypted under one session key is limited to 480k bytes.

4. Comparison

The new proposal for securing communications over ATM networks has advantages over the *ATM Security Specification Version 1.0* under the assumption that *DSA* is supported by each end system of ATM networks.

1) In the *ATM Security Specification Version 1.0*, the authentication of signaling messages for connection(s) is completed through the two-way *SME* protocol which does not support *Diffie-Hellman* (*DH*) *key agreement protocol*, because encrypting $ConfPar_a$ in *FLOW1-2WE* would require the knowledge of the secret shared key, which is not established yet. If the calling user *A* wants to encrypt $ConfPar_a$ with public key system, he needs to know the public key of the called user *B*. The public key may be obtained by retrieving *B*'s public key certificate from a public key directory or by exchanging public key certificates directly. But it will increase the number of communications. Our proposal for two-way authentication of signaling messages for connection(s) can apply *DH* to fulfill key exchange without secret shared key or *B*'s public key.

2) As far as the authentication of signaling messages for point-to-multipoint connections with key agreement in the *ATM Security Specification Version 1.0* is concerned, the calling user *A* has to encrypt $ConfPar_a$ with different public keys or secret keys and includes each encrypted $ConfPar_a$ in *FLOW1-2WE*. However, *FLOW1-2WE** of our proposal is only needed to contain $g^x \bmod p$. Therefore, *FLOW1-2WE* is much longer than our *FLOW1-2WE**.

3) Security options are completely unprotected during negotiation in the *ATM Security Specification Version 1.0*. Furthermore, they are negotiated through three-way *SME* protocol. Our proposal for security parameters negotiation protects security options from both sides with *ElGamal encryption scheme* and only needs two-way message exchange.

4) *FLOW1-3WE* of security parameter negotiation in the *ATM Security Specification Version 1.0* cannot be authenticated by the called user *B*. Therefore, a hacker can forge a lot of *FLOW1-3WE* messages with various certificates obtained over ATM networks and then send them to a called user so that the called user's server becomes slow or even shuts down. Our proposal for security parameters negotiation is immune against any hacker. Since the signature of the calling user *A* is included in *FLOW1-2WNE*,

FLOW1-2WNE can be authenticated by the called user *B*. In addition, since k_b is required to decrypt (a_a, g_a) into the significant $SecNeg_a||R_a||R_a*$ and is only known to *B*, only *B* can obtain the significant $SecNeg_a||R_a||R_a*$ from (a_a, g_a). The significant $SecNeg_a||R_a||R_a*$ means that R_a is identical to the one which sent in *FLOW1-2WE**. In this way, $SecNeg_a$ and R_a* are secretly transmitted to *B* from *A*. In the same way, $SecNeg_b$ and R_b* can be secretly transmitted to *A* from *B*.

5) The mater key and session keys for confidentiality of user exchange data in the *ATM Security Specification Version 1.0* are generated in the source and delivered from the source to the destination(s). It brings unnecessary burden of communications into ATM networks. In our proposal, on basis of one secrete shared key *K* agreed during the authentication of signaling messages for connection(s) and two more secret shared random numbers R_a* and R_b* exchanged during security parameter negotiation, the first session key in our proposal is generated by *A* and *B* individually. In addition, both *A* and *B* can compute the new session key based *K*, R_a*, R_b* and the current session key. The generation of session keys in our proposal has a particular feature: all session keys can be computed in advance and stored in memory. In one word, our proposal does not need to deliver the master key and session keys from the source to the destination(s).

6) *In the ATM Security Specification Version 1.0*, when the initiator (or the responder) wants to use a new session key, it sends key update *OAM* cells within the user data stream to exchange the new session key with the remote partner and to indicates to the remote partner when to start using the new session key. Our proposal for session key update does not need to send any *OAM* cells. During security parameters negotiation, the initiator and the responder can reach a threshold *N* for all session keys. After *N* encrypted 48-bit blocks of user data under the same session key are exchanged between the initiator and the responder, both sides synchronously update their session keys and jump to next session key.

5. Conclusion

Under the assumption that *DSA* is supported by ATM end systems, we present a new proposal for securing communications over ATM networks in this paper.

On establishing connection(s) between the initiator and the responder(s), only necessary security elements are inserted into signaling messages for connection(s) so that the responder only needs to verify signature of the initiator and perform one exponentiation operation over *GF(p)*. Therefore, the authentication of signaling messages for connection(s) can be rapidly completed.

During security parameters negotiation, security options are protected by the *ElGamal encryption scheme* and then exchanged between the initiator and the responder(s). Security options are kept secret to any adversary so that the intruder-in-the-middle attack (an intruder alters security options to request a less secure encryption method, e.g., DES with 56-bit key) can be prevented. In view of it, security parameters negotiation can be carried out in a secure way.

When providing the confidentiality of user exchange data, the session keys are generated by the master key *K* agreed during the authentication of signaling messages for connection(s) and two more negotiated security parameters. They can be computed in advance and stored in memory. Communications for delivering session key from the source to the destination(s) are avoided. In this way, negotiated security parameters are effectively used in the user data exchange.

Our proposal has three particular features: (1) key agreement protocol is carried out during the authentication of signaling messages for connection(s); (2) security options during negotiation are protected by the *ElGamal encryption scheme*; (3) session keys is updated synchronously on basis of the length of encrypted message.

Our proposal has the above mentioned advantages over the *ATM Security Specifications Version 1.0* if *DSA* are available. Under this assumption, our proposal is more secure and efficient in comparison with the *ATM Security Specifications Versions 1.0*.

6. Acknowledgements

We would like to take the opportunity to appreciate valuable comments from anonymous reviewers.

References

[1] R. Taylor and G. Findlow, "*Asynchronous Transfer Mode: Security Issues*", Proceedings of Australian Telecommunication Networks and Applications Conference, December 5-7, 1995, pp. 161-166.

[2] R. Deng, L. Gong and A. A. Lazar, "*Securing Data Transfer in Asynchronous Transfer Mode Networks*", Proceedings of GLOBECOM'95, Singapore, November 13-17, 1995, pp. 1198-1202.

[3] S. C. Chuang, "*Securing ATM Networks*", The 3rd ACM Conference on Computer and Communications Security, New Delhi, India, 1996, pp.19-30.

[4] J. Kimmins and B. Booth, "*Security for ATM networks*", Journal of Computer Security, XII(1), 1996, pp. 21-29.

[5] M. Laurent, O. Paul and P. Rolin, "*Securing Communications over ATM Networks*", IFIPSEC'97, Copenhagen, Denmark, May 1997.

[6] L. Pieson and T. Tarman, "*Requirement for Security Signaling*", ATM Forum/95-0137.

[7] M. Peyravian and E. V. Herreweghen, "*ATM Scope & Requirement*", ATM FORUM/95-0579.

[8] ATM Forum Security Working Group, "*Security Framework for ATM Networks*", ATM Forum BTD-SEC-FRWK-01.01, July 1997.

[9] ATM Forum Security Working Group, "*Phase I ATM Security Specification*", ATM Forum BTD-SEC-01.03, July 1997.

[10] D. Stevenson, N. Hillery and G. Byrd, "*Secure Communications in ATM Networks*" Communications of the ACM, Vol. 38, No. 2, February 1995, pp. 45--52.

[11] X. Yi, K. Y. Lam, Y. F. Han and Y. Gong, "*A Proposal for Securing Communications over ATM Networks*", Proceedings of International Conference on Information, Communications and Signal Processing (ICICS), September 1997 Vol. 2, pp. 631 – 634.

[12] T. D. Tarman, R. L. Hutchinson, L. G. Pierson, P. E. Sholander and E. L. Witzke, "*Algorithm – Agile Encryption in ATM Networks*", IEEE Computer, Vol.31, No.9, September 1998, pp.57-64.

[13] ATM Forum Security Working Group, "*ATM Security Specification Version 1.0*", ATM Forum AF-SEC-0100.001, February 1999.

[14] H. Leitold, R. Posch, E. Areizaga, A. Bouabdallah, M. Laurent, J. M. Mateos and O. Molino, "*Security Services in ATM Networks*", Interoperable Communication Network ICON Journal, Baltzer Science Publichers, 1999.

[15] T. ElGamal, "*A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithm*", IEEE Transactions on Information Theory, Vol.IT-31, No.4, 1985, pp.469-472.

[16] FIPS 186, "*Digital Signature Standard*", Federal Information Processing Standards Publication 186, U.S. Department of Commerce/N.I.S.T., National Technical Information Service, Springfield, Virginia, 1994.

[17] W. Diffie and M. Hellman, "*New Directions in Cryptography*", IEEE Transactions on Information Theory, Vol.IT-22, No.6, 1976, pp.644-654.

[18] ANSI X9.30 (PART 2), "*American National Standard for Financial Services – Public key cryptography using irreversible algorithms for the financial services industry – Part 2: The secure hash algorithm (SHA)*", ASC X9 Secretariat – American Bankers Association, 1993.