IJNS

International Journal of Network Security

ISSN 1816-353X (Print) Vol. 19, No. 1 (Jan. 2017)
ISSN 1816-3548 (Online)

INTERNATIONAL JOURNAL OF NETWORK SECURITY

PUBLISHING OFFICE

Min-Shiang Hwang
Department of Computer Science & Information Engineering, Asia University, Taichung 41354, Taiwan, R.O.C.
Email: mshwang@asia.edu.tw

International Journal of Network Security is published both in traditional paper form (ISSN 1816-353X) and in Internet (ISSN 1816-3548) at http://ijns.jalaxy.com.tw

PUBLISHER: Candy C. H. Lin

© Jalaxy Technology Co., Ltd., Taiwan 2005
23-75, P.O. Box, Taichung, Taiwan 40199, R.O.C.

International Journal of Network Security

Vol. 19, No. 1 (Jan. 1, 2017)

An Efficient and Secure Smart Card Based Password Authentication Scheme

Yanjun Liu[1], Chin-Chen Chang[1], and Shih-Chang Chang[2]
(Corresponding author: Chin-Chen Chang)

Department of Information Engineering and Computer Science[1]
Feng Chia University, Taichung 407, Taiwan
Department of Computer Science and Information Engineering[2]
National Chung Cheng University, Chiayi 621, Taiwan
(Email: alan3c@gmail.com)

(Received Sept. 16, 2015; revised and accepted Dec. 7 & Dec. 30, 2015)

Abstract

With the advancement of internet network technologies, remote user authentication schemes using smart cards have been widely adopted. In order to satisfy the requirements of a remote user authentication scheme, the smart card has become an essential device, one that is widely used because of its low computational cost and expedient portability. Recently, Li et al. pointed out some security weaknesses in Chen et al.'s scheme, such as forward secrecy and the wrong password login problem, and proposed an enhanced user password authentication scheme based on smart card. However, we found weaknesses in their scheme. Accordingly, we propose an enhanced scheme to remedy these security weaknesses, and prove that our scheme is more secure and efficient for network application with several positive properties.

Keywords: Authentication, hash function, security, smart card

1 Introduction

With the development of Internet network technologies, remote user authentication schemes using smart cards have been widely adopted. It is generally known that the first proposed remote authentication scheme was based on a password to identify a legitimate user even over an insecure channel [1, 12, 19]. This is the subject of a published research by Lamport in 1981 [8]. It has been claimed that there is a potential security threat caused by a stored verifier table on a remote authentication system, because the verifier table risks being modified by an adversary and has a high maintenance cost, even though all secret passwords can be encrypted against the threat of disclosure. Later, Hwang and Li [4] presented the weakness of Lamport's scheme and proposed a new scheme based on the ElGamal public-key encryption system [3] to solve the corresponding problem. In this novel method, there is no need to maintain a verifier table to achieve remote user authentication. In view of the low cost and capacity of cryptosystems, Sun [18] developed an authentication scheme to enhance the performance efficiency of Hwang and Li's scheme by involving several one-way hash operations, such that the scheme could serve as an ideal substitute for high-cost modular exponentiations. Nevertheless, these two mentioned schemes could not provide users with a free choice of passwords and mutual authentication.

Since a smart card has tamper-resistant properties, it can solve the problem of maintaining the verifier table on the server side. In a smart card based authentication system, only the user is required to hold a smart card, which was issued by the server for more convenient communication and which contained all kinds of stored secret information. Many related studies [5, 6, 7, 11, 16] have investigated smart cards and the smart card has become essential in remote authentication schemes. In 2009, Xu et al. [20] proposed a novel user authentication and claimed that their scheme is secure against various attacks. However, Song [14] and Sood et al. [15] found that Xu et al.'s scheme has some weaknesses and proposed improved schemes. Subsequently, Chen et al. [2] pointed out that there are vulnerabilities on Song and Sood et al.'s schemes. Then, Chen et al. presented an enhanced version to solve the weaknesses. Recently, Li et al. [9] claimed that Chen et al.'s scheme is still insecure and proposed a modified smart card based remote user password authentication scheme. Unfortunately, we find that there are weaknesses in Li et al.'s scheme, such as from a man-in-the-middle attack and an insider attack. Hence, we propose a novel scheme to defend against these security weaknesses. Furthermore, our proposed scheme has better computational efficiency, which has become clear by comparing our work with previous schemes. In addition, our scheme has the following properties:

F1. Mutual authentication: Both the legal user and the remote server can authenticate each other successfully.

F2. Session key agreement: The legal user and the remote server can negotiate a session key and utilize it to process subsequent communication.

F3. Freely chosen and exchanged password: A legal user can freely choose and change the password.

F4. Withstands a man-in-the-middle attack: Our scheme can withstand a man in the middle attack.

F5. Withstands an insider attack: No adversary can present an insider attack.

F6. Withstands a replay attack: No one can perform a replay attack.

F7. Perfect forward secrecy: Even if an adversary can obtain contiguous knowledge of the long-term key, the adversary cannot derive previous session keys.

F8. Satisfying known-key security: No one can utilize the secret information of a legal user to derive the session key.

The rest of this paper is organized as follows. In Section 2, we briefly review Li et al.'s smart-card-based password authentication scheme and Section 3 analyzes its weaknesses. In Section 4, we propose our scheme. Section 5 gives security and performance analyses of the proposed scheme. Finally, we present our conclusions in Section 6.

Table 1: The notations used in both Li et al.'s and our proposed schemes

U_i	The user i
S	The authentication server
ID_i	The identity of the user U_i
PW_I	The password of the user U_i
x	The master secret key of the server S
T_i	The timestamp of the user U_i
T_i'	The time of receiving the login request message
T_s	The timestamp of the server S
T_s'	The time of receiving the mutual authentication message
ΔT	A valid time threshold
$h(\cdot)$	A collision-free one-way hash function
$\|$	The message concatenation operation
$\oplus$	The bitwise XOR operation
sk	The shared session key

2 Review of Li et al.'s Scheme

In this section, we briefly review Li et al.'s smart card based password authentication scheme [9] before demonstrating its weaknesses. Their scheme is an improvement of Chen et al.'s scheme [2] and the security depends on the hardness of solving the discrete logarithm problem [13]. The notations used in both Li et al.'s and our proposed schemes are listed in Table 1.

Their scheme involves two parties, i.e., the user U_i and the server S, to communicate with each other to perform the following four phases: (1) The registration phase; (2) the login phase; (3) the authentication phase; and (4) the password change phase. Since the security basis of their scheme is the discrete logarithm problem, the server S needs to initialize some parameters before the registration phase. The server S selects two large prime numbers p and q that satisfy $p = 2q+1$, the master secret key $x \in Z_q^*$ (Z_q denotes the ring of integers modulo q and Z_q^* denotes the multiplicative group of Z_q) , and a collision-free one-way hash function $h(\cdot)$. Then, the four phases are executed as follows and are illustrated in Figure 1.

2.1 Registration Phase

Step 1. The user U_i selects his/her identity ID_i and password PW_i and submits them to the server S for registration over a secure channel.

Step 2. The server S computes two parameters: $A_i = h(ID_i \parallel PW_i)^{PW_i} \bmod p$ and $B_i = h(ID_i)^{(x+PW_i)} \bmod p$.

Step 3. The server S stores the data $\{A_i, B_i, h(\cdot), p, q\}$ on a new smart card and issues the smart card to the user U_i over a secure channel.

2.2 Login Phase

Step 1. The user U_i inserts his/her smart card into a card reader and inputs his/her identity ID_i and password PW_i.

Step 2. The smart card computes $A_i^* = h(ID_i \parallel PW_i)^{PW_i} \bmod p$ and examines whether A_i^* is equal to A_i. If the equation holds, the smart card continues to perform Step 3; otherwise, the smart card terminates this session.

Step 3. The smart card randomly selects a number $\alpha \in_R Z_q^*$ and computes the following parameters:

$$\begin{aligned} C_i &= B_i/h(ID_i)^{PW_i} \bmod p, \\ D_i &= h(ID_i)^{\alpha} \bmod p, \\ M_i &= h(ID_i \parallel C_i \parallel D_i \parallel T_i), \end{aligned}$$

where T_i is the current timestamp of the user U_i.

Step 4. The smart card sends the login request message $\{ID_i, D_i, M_i, T_i\}$ to the server S.

2.3 Authentication Phase

Step 1. The server S verifies whether ID_i is valid and $T_i^{'} - T_i \leq \Delta T$, where $T_i^{'}$ is the time of receiving the login request message and ΔT is a valid time threshold. If both conditions are true, the server S

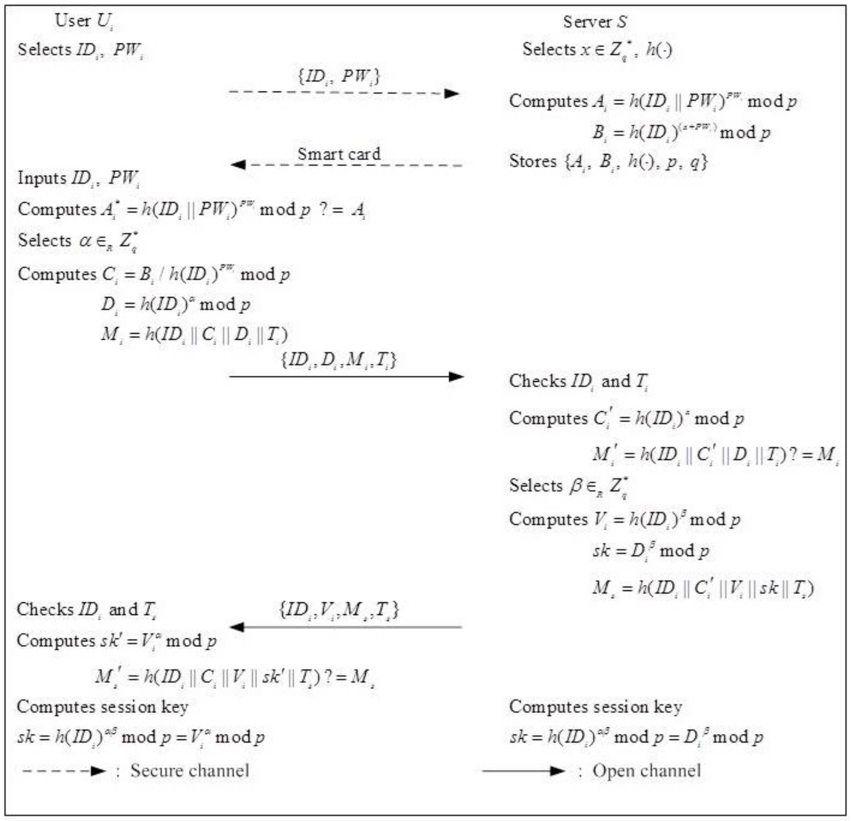

User U_i
Selects ID_i, PW_i
Server S
Selects $x \in Z_q^*$, $h(\cdot)$
$\{ID_i, PW_i\}$
Computes $A_i = h(ID_i \| PW_i)^{PW_i} \bmod p$
$B_i = h(ID_i)^{(x+PW_i)} \bmod p$
Smart card
Stores $\{A_i, B_i, h(\cdot), p, q\}$
Inputs ID_i, PW_i
Computes $A_i^* = h(ID_i \| PW_i)^{PW_i} \bmod p \ ?= A_i$
Selects $\alpha \in_R Z_q^*$
Computes $C_i = B_i / h(ID_i)^{PW_i} \bmod p$
$D_i = h(ID_i)^{\alpha} \bmod p$
$M_i = h(ID_i \| C_i \| D_i \| T_i)$
$\{ID_i, D_i, M_i, T_i\}$
Checks ID_i and T_i
Computes $C_i' = h(ID_i)^x \bmod p$
$M_i' = h(ID_i \| C_i' \| D_i \| T_i) ?= M_i$
Selects $\beta \in_R Z_q^*$
Computes $V_i = h(ID_i)^{\beta} \bmod p$
$sk = D_i^{\beta} \bmod p$
$M_s = h(ID_i \| C_i' \| V_i \| sk \| T_s)$
Checks ID_i and T_s
$\{ID_i, V_i, M_s, T_s\}$
Computes $sk' = V_i^{\alpha} \bmod p$
$M_s' = h(ID_i \| C_i \| V_i \| sk' \| T_s) ?= M_s$
Computes session key
$sk = h(ID_i)^{\alpha\beta} \bmod p = V_i^{\alpha} \bmod p$
Computes session key
$sk = h(ID_i)^{\alpha\beta} \bmod p = D_i^{\beta} \bmod p$
: Secure channel
: Open channel

Figure 1: Li et al.'s scheme

continues to execute Step 2; otherwise, the server S rejects the login request.

Step 2. The server S computes two parameters: $C_i' = h(ID_i)^x \bmod p$ and $M_i' = h(ID_i \parallel C_i' \parallel D_i \parallel T_i)$ Then, the server S compares whether M_i' equals M_i. If they are equal, the server S confirms that the user U_i is valid and the login request is accepted; otherwise, the login request is rejected.

Step 3. The server S randomly selects a number $\beta \in_R Z_q^*$ and computes the following parameters:

$$\begin{aligned} V_i &= h(ID_i)^{\beta} \bmod p, \\ sk &= D_i^{\beta} \bmod p, \\ M_s &= h(ID_i \parallel C_i' \parallel V_i \parallel sk \parallel T_s), \end{aligned}$$

where T_s is the current timestamp of the server S.

Step 4. The server S sends the mutual authentication message $\{ID_i, V_i, M_s, T_s\}$ to the user U_i.

Step 5. Upon receiving the message $\{ID_i, V_i, M_s, T_s\}$, the user U_i checks the validity of ID_i and whether $T_s' - T_s \leq \Delta T$, where T_s' is the time of receiving the mutual authentication message. If both of them hold, the user U_i continues to perform Step 6; otherwise, the user U_i terminates this connection.

Step 6. The user U_i computes two parameters: $sk' = V_i^{\alpha} \bmod p$ and $M_s' = h(ID_i \parallel C_i \parallel V_i \parallel sk' \parallel T_s)$. Then, the user U_i checks whether M_s' equals M_s. If they are equal, the validity of the server S is authenticated; otherwise, the session is terminated.

Step 7. The user U_i and the server S construct a shared session key $sk = h(ID_i)^{\alpha\beta} \bmod p$ to ensure the secret communication.

2.4 Password Change Phase

Step 1. The user U_i inserts his/her smart card into a card reader, enters his/her old identity ID_i and password PW_i, and requests to change the password.

Step 2. The smart card computes $A_i^* = h(ID_i \parallel PW_i)^{PW_i} \bmod p$ and checks whether A_i^* equals A_i that is stored in the smart card. If the equation holds, the user U_i submits the new password PW_i^{new}; otherwise, the smart card rejects the password change request.

Step 3. The smart card computes $A_i^{new} = h(ID_i \parallel PW_i^{new})^{PW_i^{new}} \bmod p$ and $B_i^{new} = B_i \cdot h(ID_i)^{PW_i^{new}} / h(ID_i)^{PW_i} \bmod p$. Then, the smart card replaces A_i and B_i with A_i^{new} and B_i^{new}, respectively.

3 Weaknesses of Li et al.'s Scheme

Li et al.'s scheme [9] can correct the design flaws of Chen et al.'s scheme [2], such as by ensuring perfect forward secrecy, quickly detecting the wrong password via the smart card without interacting with the server in the login phase, and provides a friendly and efficient password change. Additionally, Li et al. claimed that their scheme is very secure and can resist various types of attacks. We found, however, that Li et al.'s scheme cannot withstand a man-in-the-middle attack or an insider attack. In addition, the use of modulus exponential operations incurs a considerable computational cost. The details of these weaknesses are discussed below.

3.1 Man-in-the-middle Attack

Li et al.'s scheme is vulnerable to a man-in-the-middle attack. Suppose that there exists an attacker U_E between the user U_i and the server S. The attacker U_E can intercept the login request message and the mutual authentication message transmitted between the user U_i and the server S, and then modify these messages. U_E can act as the user U_i to communicate with the server S and act as the server S to communicate with the user U_i without detection. This type of attack can be described as follows:

Step 1. In the login phase, the user U_i's smart card sends the login request message $\{ID_i, D_i, M_i, T_i\}$ to the server S. The attacker U_E intercepts this message.

Step 2. Since $M_i = h(ID_i \parallel C_i \parallel D_i \parallel T_i)$, the attacker U_E uses the intercepted values of ID_i, D_i, T_i, and M_i to guess C_i Due to the fact that $C_i = B_i/h(ID_i)^{PW_i} \bmod p = h(ID_i)^x \bmod p$ would remain the same in different sessions of the user U_i and the server S, the attacker U_E can easily determine the value of C_i.

Step 3. The attacker U_E generates $D_E = h(ID_i)^e$ and computes $M_E = h(ID_i \parallel C_i \parallel D_E \parallel T_E)$. After that, U_E sends $\{ID_i, D_E, M_E, T_E\}$ to the server S.

Step 4. The server S first checks the validity of ID_i and T_E, and then computes $C_i^{'} = h(ID_i)^x \bmod p$ and $M_E^{'} = h(ID_i \parallel C_i^{'} \parallel D_E \parallel T_E)$ Afterwards, the server S compares whether $M_E^{'}$ equals M_E. If they are equal, the server S believes that the attacker U_E is authenticated as the user U_i.

Step 5. The server S computes $V_i = h(ID_i)^{\beta} \bmod p$, $sk = D_E^{\beta} \bmod p = h(ID_i)^{e\beta} \bmod p$, and $M_s = h(ID_i \parallel C_i^{'} \parallel V_i \parallel sk \parallel T_s)$, then sends the mutual authentication message $\{ID_i, V_i, M_s, T_s\}$ to the user U_i. The attacker U_E intercepts this message.

Step 6. The attacker U_E generates $V_E = h(ID_i)^e$, $sk_E = V_E \cdot D_i = h(ID_i)^{e\alpha} \bmod p$ and $M_E^{''} = h(ID_i \parallel C_i^{'} \parallel V_E \parallel sk_E \parallel T_E^{'})$. After that, U_E sends $\{ID_i, V_E, M_E^{''}, T_E^{'}\}$ to the user U_i.

Step 7. The user U_i first checks the validity of ID_i and $T_E^{'}$, and then computes $sk^{'} = V_E^{\alpha} \bmod p = h(ID_i)^{e\alpha} \bmod p$ and $M_E^{'''} = h(ID_i \parallel C_i \parallel V_E \parallel sk^{'} \parallel T_E^{'})$ Afterwards, the user U_i checks whether $M_E^{'''}$ equals $M_E^{''}$. If they are equal, the user U_i believes that the attacker U_E is authenticated as the server S.

After performing the authentication phase, the user U_i believes that the attacker U_E is the server S and the server S believes that the attacker U_E is the user U_i. Moreover, user U_i and the server S trust that they have established a common session key. However, server S and the attacker U_E share a session key $sk = h(ID_i)^{e\beta} \bmod p$; and user U_i and the attacker U_E share another session key $sk^{'} = h(ID_i)^{e\alpha} \bmod p$. Consequently, Li et al.'s scheme cannot prevent a man-in-the-middle attack.

3.2 Insider Attack

If server S directly obtains user U_i's password PW_i, an insider attack takes place when an intruder steals PW_i from S. In Li et al.'s scheme, the user U_i selects their password PW_i and submits it to the server S for registration over a secure channel. Therefore, server S can obtain the user U_i's password PW_i and cannot withstand an insider attack.

3.3 Computational Inefficiency

From Li et al.'s scheme, we can see that it uses too many modulus exponential operations, which can incur unnecessary overhead. The computational cost in the login and authenticated phases are $3E + 1M + 3H$ and $4E + 4H$, respectively, where E is modulus exponential operations, M is multiplication/division operations, and H is hashing operations. Li et al. claimed that although their scheme requires a higher computational cost, it can achieve higher security and usability compared with other related schemes. Unfortunately, this is not true according to the discussion in Subsections 3.1 and 3.2. In fact, the modulus exponential operations can be replaced with other appropriate operations to reduce the computational cost.

4 Our Proposed Scheme

To overcome the aforementioned weaknesses, we propose a novel smart card based password authentication scheme, which is secure and more efficient. By using the combination of collision-free one-way hash functions, bitwise XOR ($\oplus$) and concatenation ($\parallel$) operations instead of modulus exponential operations, our proposed scheme can significantly enhance computational efficiency while satisfying various security requirements. Our proposed scheme consists of four phases: (1) The registration phase; (2) the login phase; (3) the authentication phase; and (4) the password change phase. In the following, we will describe the proposed scheme in detail.

4.1 Registration Phase

At the beginning of our proposed scheme, the server S selects the master secret key x and a collision-free one-way hash function $h(\cdot)$. Then, the user U_i registers to the server S by the way below:

Step 1. The user U_i first selects his/her identity ID_i, password PW_i, and a random number r, and then computes $h(r \parallel PW_i)$. U_i submits $\{ID_i, h(r \parallel PW_i)\}$ to the server S for registration over a secure channel.

Step 2. The server S computes the following parameters:

$$\begin{aligned} A_i &= h(ID_i \oplus x) \parallel h(x), \\ B_i &= A_i \oplus h(r \parallel PW_i), \\ C_i &= h(A_i \parallel ID_i \parallel h(r \parallel PW_i)). \end{aligned}$$

Step 3. The server S stores the data $\{B_i, C_i, h(\cdot)\}$ on a new smart card and issues the smart card to the user U_i over a secure channel.

Step 4. The user U_i stores the random number r into the smart card.

The registration phase is depicted in Figure 2.

4.2 Login Phase

This phase is invoked whenever the user U_i wants to login to the server S. The steps of this phase are conducted as follows:

Step 1. The user U_i inserts his/her smart card into a card reader and inputs his/her identity ID_i and password PW_i.

Step 2. The smart card first computes two parameters: $A_i^{'} = B_i \oplus h(r \parallel PW_i)$ and $C_i^{'} = h(A_i^{'} \parallel ID_i \parallel h(r \parallel PW_i))$. Then, the smart card examines whether $C_i^{'}$ is equal to C_i. If the equation holds, the smart card continues to perform Step 3; otherwise, the smart card terminates this session.

Step 3. The smart card randomly selects a number α and computes the following parameters:

$$\begin{aligned} D_i &= h(ID_i \oplus \alpha), \\ E_i &= A_i^{'} \oplus \alpha \oplus T_i, \end{aligned}$$

where T is the current timestamp of the user U_i.

Step 4. The smart card sends the login request message $\{ID_i, D_i, E_i, T_i\}$ to the server S.

4.3 Authentication Phase

After completing this phase, the user U_i and the server S can mutually authenticate each other and establish a shared session key for the subsequent secret communication. The steps of this phase are shown as follows:

Step 1. The server S verifies whether ID_i is valid and $T_i^{'} - T_i \leq \Delta T$, where $T_i^{'}$ is the time of receiving the login request message and ΔT is a valid time threshold. If both conditions are true, the server S continues to execute Step 2; otherwise, the server S rejects the login request.

Step 2. The server S computes the following parameters:

$$\begin{aligned} A_i &= h(ID_i \oplus x) \parallel h(x), \\ \alpha^{'} &= E_i \oplus A_i \oplus T_i, \\ D_i^{'} &= h(ID_i \oplus \alpha^{'}). \end{aligned}$$

Then, the server S compares whether $D_i^{'}$ equals D_i. If they are equal, the server S confirms that the user U_i is valid and the login request is accepted; otherwise, the login request is rejected.

Step 3. The server S randomly selects a number β and computes the following parameters:

$$\begin{aligned} F_i &= h(ID_i \oplus \beta), \\ G_i &= A_i \oplus \beta \oplus T_s, \end{aligned}$$

where T_s is the current timestamp of the server S.

Step 4. The server S sends the mutual authentication message $\{F_i, G_i, T_s\}$ to the user U_i.

Step 5. Upon receiving the message $\{F_i, G_i, T_s\}$, the user U_i checks the validity of T_s. If $T_s^{'} - T_s \leq \Delta T$, where $T_s^{'}$ is the time of receiving the mutual authentication message, the user U_i continues to perform Step 6; otherwise, the user U_i terminates this connection.

Step 6. The user U_i computes $\beta^{'} = G_i \oplus A_i^{'} \oplus T_s$ and $F_i^{'} = h(ID_i \oplus \beta^{'})$, and then checks whether $F_i^{'}$ equals F_i. If they are equal, the validity of the server S is authenticated; otherwise, the session is terminated.

Step 7. The user U_i and the server S construct a shared session key $sk = h(\alpha \parallel \beta^{'} \parallel h(A_i^{'} \oplus ID_i)) = h(\alpha^{'} \parallel \beta \parallel h(A_i \oplus ID_i))$ to ensure the secret communication.

The login and authentication phases are shown in Figure 3.

4.4 Password Change Phase

Step 1. The user U_i inserts his/her smart card into a card reader, enters his/her old identity ID_i and password PW_i, and requests to change the password.

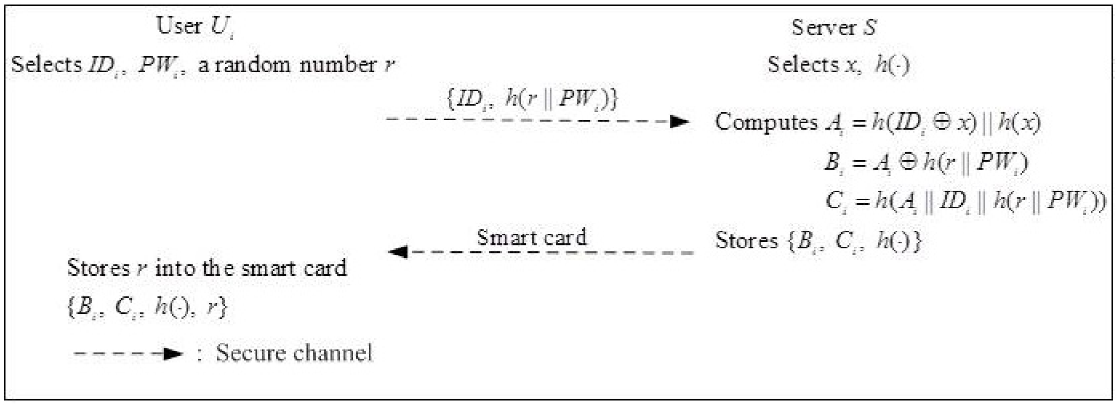

User U_i
Selects ID_i, PW_i, a random number r
Server S
Selects x, $h(\cdot)$
$\{ID_i, h(r \| PW_i)\}$
Computes $A_i = h(ID_i \oplus x) \| h(x)$
$B_i = A_i \oplus h(r \| PW_i)$
$C_i = h(A_i \| ID_i \| h(r \| PW_i))$
Smart card
Stores $\{B_i, C_i, h(\cdot)\}$
Stores r into the smart card
$\{B_i, C_i, h(\cdot), r\}$
: Secure channel

Figure 2: Registration phase of our proposed scheme

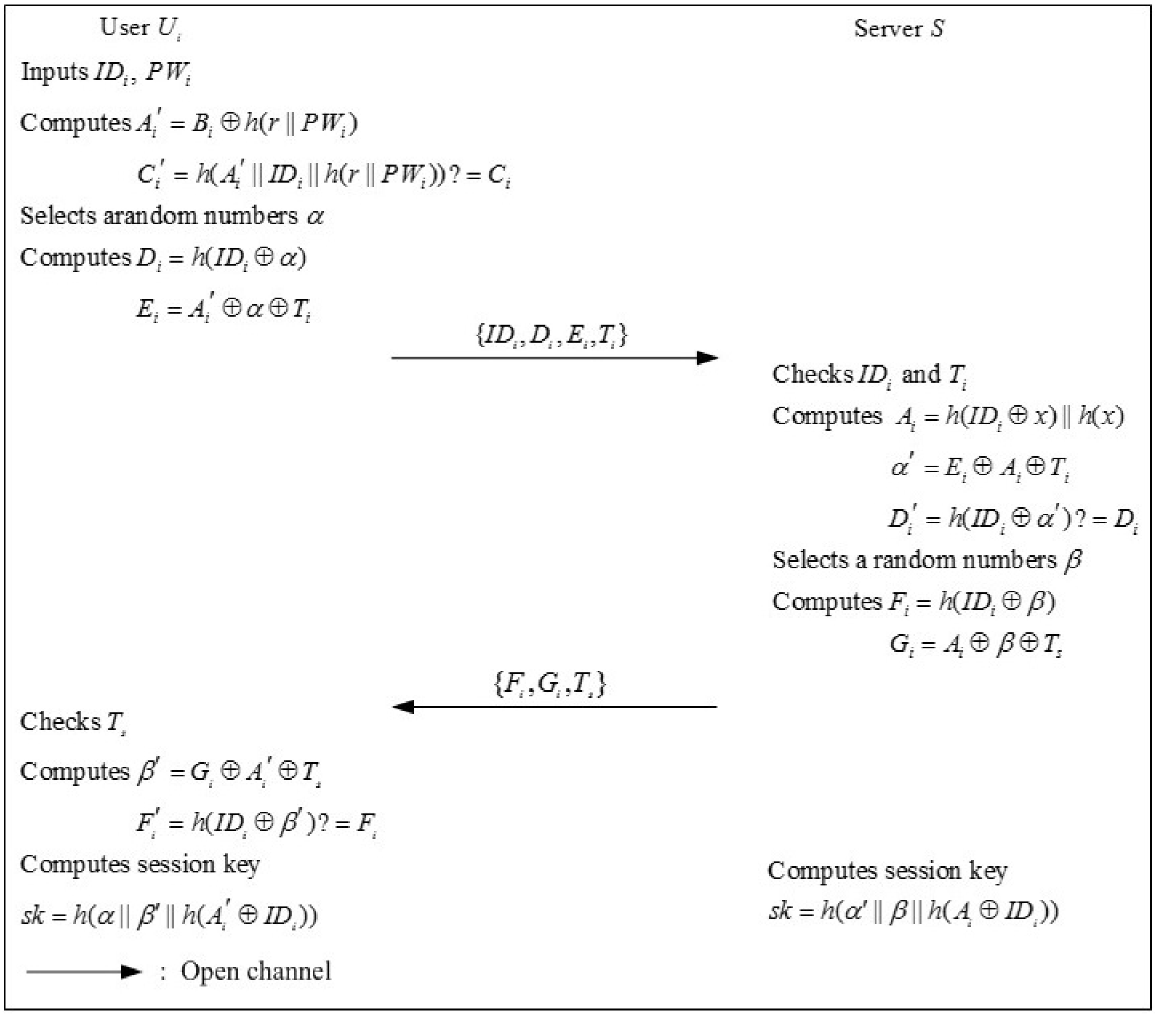

User U_i
Server S
Inputs ID_i, PW_i
Computes $A_i' = B_i \oplus h(r \| PW_i)$
$C_i' = h(A_i' \| ID_i \| h(r \| PW_i)) ? = C_i$
Selects arandom numbers α
Computes $D_i = h(ID_i \oplus \alpha)$
$E_i = A_i' \oplus \alpha \oplus T_i$
$\{ID_i, D_i, E_i, T_i\}$
Checks ID_i and T_i
Computes $A_i = h(ID_i \oplus x) \| h(x)$
$\alpha' = E_i \oplus A_i \oplus T_i$
$D_i' = h(ID_i \oplus \alpha') ? = D_i$
Selects a random numbers β
Computes $F_i = h(ID_i \oplus \beta)$
$G_i = A_i \oplus \beta \oplus T_s$
$\{F_i, G_i, T_s\}$
Checks T_s
Computes $\beta' = G_i \oplus A_i' \oplus T_s$
$F_i' = h(ID_i \oplus \beta') ? = F_i$
Computes session key
$sk = h(\alpha \| \beta' \| h(A_i' \oplus ID_i))$
Computes session key
$sk = h(\alpha' \| \beta \| h(A_i \oplus ID_i))$
: Open channel

Figure 3: Login and authentication phases of our proposed scheme

Step 2. The smart card computes $A_i^* = B_i \oplus h(r \parallel PW_i)$ and $C_i^* = h(A_i^* \parallel ID_i \parallel h(r \parallel PW_i))$, and then checks whether C_i^* equals C_i that is stored in the smart card. If the equation holds, the user U_i submits the new password PW_i^{new}; otherwise, the smart card rejects the password change request.

Step 3. The smart card computes $B_i^{new} = A_i^* \oplus h(r \parallel PW_i^{new})$ and $C_i^{new} = h(A_i^* \parallel ID_i \parallel h(r \parallel PW_i^{new}))$. Then, the smart card replaces B_i and C_i with B_i^{new} and C_i^{new}, respectively.

5 Analysis of the Proposed Scheme

In this section, we analyze the security and performance of our proposed scheme and make comparisons with other related works.

5.1 Functionality and Security Analyses

5.1.1 Mutual Authentication

Our proposed scheme can achieve mutual authentication such that the user and the server can successfully verify the validity of each other. In Step 2 of the authentication phase, the server S computes $D_i^{'} = h(ID_i \oplus \alpha^{'})$, and then compares whether $D_i^{'}$ equals $D_i^{'}$ that was sent by the user U_i. If they are equal, the server S confirms that user U_i is valid. On the other hand, in Step 6 of the authentication phase, user U_i computes $F_i^{'} = h(ID_i \oplus \beta^{'})$, and then checks whether $F_i^{'}$ equals F_i that was sent by the server S. If they are equal, the validity of the server S is authenticated.

5.1.2 Session Key Agreement

After achieving mutual authentication, the user and the server must negotiate a common session key, which is used to encrypt the data transmitted between the user and the server in the subsequent confidential communications. In our proposed scheme, the user and the server share the session key $sk = h(\alpha \parallel \beta^{'} \parallel h(A_i^{'} \oplus ID_i)) = h(\alpha^{'} \parallel \beta \parallel h(A_i \oplus ID_i))$ at the end of the authentication phase.

5.1.3 Freely Chosen and Exchanged Password

Our proposed scheme allows each user to choose their password in the registration phase so that users can easily remember their passwords. In addition, each user can change their password in the password change phase. If user U_i wants to update their password, the smart card checks the validity of the old password by comparing whether C_i^* equals C_i. If so, user U_i submits the new password PW_i^{new}. The smart card uses PW_i^{new} to compute B_i^{new} and C_i^{new}, and then replaces B_i and C_i with B_i^{new} and C_i^{new}, respectively. The password change phase is friendly and efficient since the smart card can complete both the tasks of verification of old passwords and updating of new passwords. Thus, the user does not need to communicate with the server to change the password.

5.1.4 Withstanding a Man-in-the-middle Attack

Assume that there exists an attacker U_E between the user U_i and the server S. In the login phase, the attacker U_E can intercept the login request message $\{ID_i, D_i, E_i, T_i\}$ and attempts to forge it to act as user U_i. However, U_E cannot get $A_i^{'}$ and α from the intercepted message. So, if U_E generates a fake E_i and sends it to the server S, S can check that $D_i^{'}$ is not equal to the received D_i and concludes that U_E is not a valid user. On the other hand, the attacker U_E can intercept the mutual authentication message $\{F_i, G_i, T_s\}$ and wants to forge it to act as the server S. Similarly, because U_E cannot obtain A_i and β, the user U_i will not be mislead by the forged $F_i^{'}$ and concludes that U_E is not a valid server. Therefore, attacker U_E cannot modify the messages to pass the login and the authentication phases. This indicates that our proposed scheme can prevent a man-in-the-middle attack.

5.1.5 Withstanding an Insider Attack

In the registration phase, the user conceals the password in a ciphertext from the server to resist an insider attack. More specifically, user U_i first selects their password PW_i and a random number r, and then submits $h(r \parallel PW_i)$ to the server S for registration over a secure channel. As a result, server S cannot get the correct password PW_i and an insider attack will not occur.

5.1.6 Withstanding Replay Attack

A replay attack means a malicious intruder repeats or delays valid transmitted messages without detection. Our proposed scheme can resist a replay attack by utilizing timestamps in the login and authentication phases. In Step 4 of the login phase, the smart card adds the timestamp T_i into the login request message $\{ID_i, D_i, E_i, T_i\}$ and sends it to the server S. Meanwhile, in Step 4 of the authentication phase, the server S puts the timestamp T_s into the mutual authentication message $\{F_i, G_i, T_s\}$ and conveys it to the user U_i. Therefore, the user U_i and the server S can verify the occurrence of a replay attack by checking timestamps T_i and T_s.

5.1.7 Providing Perfect Forward Secrecy

Perfect forward secrecy can ensure that any previously established session keys are not disclosed to the attacker even if the server's master secret key is compromised. In our proposed scheme, the shared session key $sk = h(\alpha \parallel \beta^{'} \parallel h(A_i^{'} \oplus ID_i)) = h(\alpha^{'} \parallel \beta \parallel h(A_i \oplus ID_i))$, where $\alpha = \alpha^{'}$, $\beta = \beta^{'}$, $A_i = h(ID_i \oplus x) \parallel h(x)$ and $A_i^{'} = B_i \oplus h(r \parallel PW_i)$. Suppose that an attacker obtained the server's master secret key x. If the attacker wants to derive the previous session key sk, they must know α

Table 2: Functionality comparison of our scheme and other related schemes

	F1	F2	F3	F4	F5	F6	F7	F8
Juang et al. [6]	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Song [14]	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Chen et al. [2]	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Li et al. [9]	Yes	Yes	Yes	No	No	Yes	Yes	Yes
Sun et al. [17]	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
Li et al. [10]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Our scheme	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

F1: Mutual authentication; F2: Session key agreement; F3: Freely chosen and exchanged password; F4: Withstanding man in the middle attack; F5: Withstanding insider attack; F6: Withstanding replay attack; F7: Providing perfect forward secrecy; F8: Satisfying known-key security.

and β. However, α and β are not directly transmitted between user U_i and server S via the public channel, but are encrypted into the ciphertext D_i and F_i, respectively. Therefore, α and β cannot be obtained by the attacker, which implies that our proposed scheme provides perfect forward secrecy.

5.1.8 Satisfying Known-key Security

Known-key security guarantees that other session keys will not be derived by the attacker from the compromised session key. Our proposed scheme can satisfy known-key security by allowing user U_i and the server S to establish unique session keys in their different login and authentication phases. Assume that a session key $sk = h(\alpha \parallel \beta \parallel h(A_i \oplus ID_i))$ is compromised. Since α and β are random numbers selected by the user U_i and the server S, respectively, different values of α and β will be selected in different sessions. As a result, even if the attacker gets sk, α, and β, they cannot compute another session key $sk^{'}$ from the compromised sk without knowing $\alpha^{'}$ and $\beta^{'}$ from the other sessions. Therefore, our proposed scheme can satisfy the known-key security problem.

The functionality comparison of our proposed scheme with other related works [2, 6, 9, 10, 14, 17] is summarized in Table 2, which infers that our proposed scheme is more secure and practical than other related works.

5.2 Performance Analysis

In this subsection, we evaluate the performance of our proposed scheme in terms of computational cost. Table 3 compares the computational cost of our proposed scheme and other related schemes [2, 6, 9, 10, 14, 17]. From Table 3, we can see that all of other existing schemes involve some time-consuming operations, such as modulus exponential operations, symmetric encryption/decryption operations or multiplication/division operations. In particular, among these three operations, multiplication/division operations are faster than symmetric encryption/decryption operations while symmetric encryption/decryption operations are faster than modulus exponential operations. Fortunately, our proposed scheme only utilizes one-way hash functions, which are much faster than the mentioned three operations. Therefore, this method can significantly enhance computational efficiency while retaining higher security as shown in Table 2.

6 Conclusions

In this paper, we proposed a smart card based password authentication scheme to overcome the security weaknesses of Li et al.'s scheme. Our proposed scheme can achieve mutual authentication and users can freely choose and change their passwords. We prove that our proposed scheme can resist various types of attack, such as a man-in-the-middle attack, insider attack, and replay attack. Furthermore, our proposed scheme has better computational efficiency than other related works.

References

[1] C. C. Chang, C. Y. Lee and Y. C. Chiu, "Enhanced authentication scheme with anonymity for roaming service in global mobility networks," *Computer Communications*, vol. 32, no. 4, pp. 611–618, 2009.

[2] B. L. Chen, W. C. Kuo and L. C. Wuu, "Robust smart-card-based remote user password authentication scheme," *International Journal of Communication Systems, in press.* (`http://dx.doi.org/10.1002/dac.2368`)

[3] T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, 1985.

[4] M. S. Hwang and L. H. Li, "A new remote user authentication scheme using smart card," *IEEE Trans-*

Table 3: Computational cost comparison of our scheme and other related schemes

	C1	C2	C3	C4	C5	C6
Juang et al. [6]	1H	2H+3S	3H+3S	4H+6S+1M	1H+5S	1H+5S
Song [14]	-	2H+E	3H+1S	3H+1S+1E	Null	Null
Chen et al. [2]	-	1H+1E	2H+2M+4E	H+M+4E	3H+2M+2E	3H+2M+3E
Li et al. [9]	-	2H+2E	4H+M+4E	3H+3E	3H+2M+4E	-
Sun et al. [17]	-	2H+1S	4H+2M	4H+1S+2M	2H	-
Li et al. [10]	1H	2H+3S	8H+4S	10H+10S+1M	1H+6S	1H+9S
Our scheme	1H	3H	6H	6H	4H	-

C1: Computational cost of the user in registration phase; C2: Computational cost of the server in registration phase; C3: Computational cost of the user in login and authentication phases; C4: Computational cost of the server in login and authentication phases; C5: Computational cost of the user in password change phase; C6: Computational cost of the server in password change phase; H: Hashing operation; E: Modulus exponential operation; S: Symmetric encryption/decryption operation; M: Multiplication/division operation; Null: Cannot provide this functionality.

actions on Consumer Electronics, vol. 46, no. 1, pp. 28–30, 2000.

[5] W. S. Juang, "Efficient multi-server password authenticated key agreement using smart cards," *IEEE Transactions on Consumer Electronics*, vol. 50, no. 1, pp. 251–255, 2004.

[6] W. S. Juang, S. T. Chen and H. T. Liaw, "Robust and efficient password-authenticated key agreement using smart card," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 6, pp. 2551–2556, 2008.

[7] S. K. Kim and M. G. Chung, "More secure remote user authentication scheme," *Computer Communications*, vol. 32, no. 6, pp. 1018–1021, 2009.

[8] L. Lamport, "Password authentication with insecure communication," *Communications of the ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[9] X. Li, J. Niu, M. K. Khan, and J. Liao, "An enhanced smart card based remote user password authentication scheme," *Journal of Network and Computer Applications, in press.* (`http://dx.doi.org/10.1016/j.jnca.2013.02.034.`)

[10] X. X. Li, W. D. Qiu, D. Zheng, K. F. Chen, and J. H. Li, "Anonymity enhancement on robust and efficient password-authenticated key agreement using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 2, pp. 793–800, 2010.

[11] J. Y. Liu, A. M. Zhou, and M. X. Gao, "A new mutual authentication scheme based on nonce and smart card," *Computer Communications*, vol. 31, no. 10, pp. 2205–2209, 2008.

[12] M. Peyravian and N. Zunic, "Methods for protecting password transmission," *Computer and Security*, vol. 19, no. 5, pp. 466–469, 2006.

[13] B. Schneier, *Applied Cryptography (2nd Edition)*, New York: Wiley, 1996.

[14] R. Song, "Advanced smart card based password authentication protocol," *Computer Standards and Interfaces*, vol. 32, no. 5, pp. 321–325, 2010.

[15] S. K. Sood, A. K. Sarje, and K. Singh, "An improvement of xu et al.'s authentication scheme using smart cards," in *Proceedings of the Third Annual ACM Bangalore Conference*, pp. 17–22, Bangalore, Karnataka, India, 2010.

[16] D. Z. Sun, J. P. Huai, J. Z. Sun, and J. X. Li, "Cryptanalysis of a mutual authentication scheme based on nonce and smart cards," *Computer Communications*, vol. 32, no. 6, pp. 1015–1017, 2009.

[17] D. Z. Sun, J. P. Huai, J. Z. Sun, J. X. Li, J. W. Zhang, and Z. Y. Feng, "Improvements of juang et al.'s password-authenticated key agreement scheme using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 6, pp. 2284–2291, 2009.

[18] H. M. Sun, "An efficient remote user authentication scheme using smart cards," *IEEE Transactions on Consumer Electronics*, vol. 46, no. 4, pp. 958–961, 2000.

[19] T. C. Wu and H. S. Sung, "Authentication passwords over an insecure channel," *Computer and Security*, vol. 15, no. 5, pp. 431–439, 1996.

[20] J. Xu, W. T. Zhu, and D. G. Feng, "An improved smart card based password authentication scheme with provable security," *Computer Standards and Interfaces*, vol. 31, no. 4, pp. 723–728, 2009.

Yanjun Liu received her Ph.D. degree in 2010, in School of Computer Science and Technology from University of Science and Technology of China (USTC), Hefei, China. She is currently a postdoctor at Feng Chia University, Taichung, Taiwan. Her current research interests include information security and computer cryptography.

Chin-Chen Chang received his Ph.D. degree in computer engineering from National Chiao Tung University. His first degree is Bachelor of Science in Applied Mathematics and master degree is Master of Science in computer and decision sciences. Both were awarded in National Tsing Hua University. Dr. Chang served in National Chung Cheng University from 1989 to 2005. His current title is Chair Professor in Department of Information Engineering and Computer Science, Feng Chia University, from Feb. 2005. Prior to joining Feng Chia

University, Professor Chang was an associate professor in Chiao Tung University, professor in National Chung Hsing University, chair professor in National Chung Cheng University. He had also been Visiting Researcher and Visiting Scientist to Tokyo University and Kyoto University, Japan. During his service in Chung Cheng, Professor Chang served as Chairman of the Institute of Computer Science and Information Engineering, Dean of College of Engineering, Provost and then Acting President of Chung Cheng University and Director of Advisory Office in Ministry of Education, Taiwan. Professor Chang has won many research awards and honorary positions by and in prestigious organizations both nationally and internationally. He is currently a Fellow of IEEE and a Fellow of IEE, UK. And since his early years of career development, he consecutively won Outstanding Talent in Information Sciences of the R. O. C., AceR Dragon Award of the Ten Most Outstanding Talents, Outstanding Scholar Award of the R. O. C., Outstanding Engineering Professor Award of the R. O. C., Distinguished Research Awards of National Science Council of the R. O. C., Top Fifteen Scholars in Systems and Software Engineering of the Journal of Systems and Software, and so on. On numerous occasions, he was invited to serve as Visiting Professor, Chair Professor, Honorary Professor, Honorary Director, Honorary Chairman, Distinguished Alumnus, Distinguished Researcher, Research Fellow by universities and research institutes. His current research interests include database design, computer cryptography, image compression and data structures.

Shih-Chang Chang received his B.S. degree in 2005 and his M.S. degree in 2007, both in Department of Information Engineering and Computer Science from Feng Chia University, Taichung, Taiwan. He is currently pursuing his Ph.D. degree in Computer Science and Information Engineering from National Chung Cheng University, Chiayi, Taiwan. His current research interests include electronic commerce, information security, computer cryptography, and mobile communications.

A Homomorphic Universal Re-encryptor for Identity-based Encryption

Liang Liu[1] and Jun Ye[2,3]
(Corresponding author: Jun Ye)

State Key Laboratory of Integrated Service Networks (ISN), Xidian University[1]
No. 2 South Taibai Road, Yanta District, Xi'an, Shaanxi 710071, China
Artificial Intelligence Key Laboratory of Sichuan Province, Sichuan University of Science & Engineering[2]
Sichuan Province University Key Laboratory of Bridge Non-destruction, Detecting and Engineering Computing[3]
No. 180, School Street, Huixing Road, Sichuan 643000, China
(Email: yejun@suse.edu.cn)

(Received Oct. 23, 2015; revised and accepted Dec. 7 & Dec. 30, 2015)

Abstract

Re-encryption (or proxy re-encryption) is a very useful cryptographic primitive which is able to transform a ciphertext under one public key into a new ciphertext encrypting the same message but under another different public key. It plays an important role in modern secure communication and information exchange via various kinds of network infrastructure. In addition to traditional public-key encryption scheme, re-encryption can also come into force in other cryptosystems like Identity-Based Encryption (IBE) and more advanced Functional Encryption (FE), making the enhanced schemes more powerful as well as easy-to-use. In this work, we have proposed a novel identity-based proxy re-encryption (IBPRE) scheme which to the maximum extent reduces the workloads in the user side by delivering the re-encryption key (RK) generation work to the proxy server. Besides, it is likewise able to prevent possible bottlenecks for the users, like re-encryption key management.

Keywords: Fully homomorphic encryption, identity-based encryption, proxy re-encryption, re-encryptor

1 Introduction

Cryptographic primitives supporting intermediate transformations from one object (ciphertext or signature) to another without leakage of sensitive information have found their irreplaceable places in modern Internet era. The most common application of these primitives may be proxy re-encryption in e-mail relay. A senior manager Alice wants to forward an encrypted e-mail from the executive level of the enterprise to her subordinate Bob. Of course, she is able to decrypt the encrypted e-mail by her private key SK_{Alice} and then encrypts the plaintext under Bob's public key PK_{Bob} to obtain the corresponding encrypted e-mail, which will then be sent to Bob. On the surface, it seems that this method actually achieves the goal. However, we argue that this trivial solution results in several shortcomings. The most obvious point is that the initial ciphertext owner Alice must execute all these computations herself - including decryption and encryption - to produce the ciphertext for Bob. In some scenarios, these computation workloads are awfully cumbersome.

Another way to solve the above mentioned issue may be to introduce a proxy to accomplish those transformation workloads on behalf of the manager Alice, who has usually been named as *delegator* in the proxy re-encryption/re-signature scheme. Accordingly, the term for the proxy is *delegatee*, the role of which is commonly played by a more powerful server. Then if the delegator Alice wants to largely reduce her computation workloads caused by decryption and encryption, she can generate a re-encryption key $RK_{\text{Alice}\to\text{Bob}}$ by which anyone is capable of transforming a ciphertext c_{Alice} under her public key to the corresponding ciphertext c_{Bob} under Bob's public key without any plaintext leakage or private key infringement. Nevertheless, this approach also suffers from some problems, which we will detail later.

Identity-based encryption is a useful as well as powerful primitive envisioned by Shamir [24] in 1984. But due to the lag of mathematical tools, the first scheme based on bilinear maps was proposed by Dan Boneh [2] after 17 years. IBE is more natural and convenient for the system users because when one user U_1 wishes to encrypt a message to another user U_2, she needs not to know the public key of U_2. In contrast, she can just encrypt the message she wishes to send under a publicly accessible identity corresponding to U_2. This tremendously cuts down the work from the management of user certificates. So extending the re-encryption techniques to identity-based encryption schemes and other advanced encryption schemes [3, 11] is reasonable and necessary.

1.1 Motivation

In the traditional research works on (proxy) re-encryption from former literatures, researchers use re-encryption in such a scenario: Alice and Bob are legal users within a cryptographic system with their own public-private key pairs. In the real-life scenario, this cryptography system may be used for an e-mail server or a cloud storage service provider. Alice wants to grant her e-mail server some privilege to forward e-mails to Bob for her convenience, otherwise every time she must decrypt the e-mail and then encrypt it under Bob's public key/identity. As we have stated above, re-encryption is essential in this scenario.

But unfortunately, in all of the former literatures, the authors only consider the scenarios in which re-encryptions happen between only two users. Although some schemes support multi-use (multi-hop) re-encryption, the inputs for generating re-encryption keys are not the same. For instance, the inputs for generating $RK_{U_1 \to U_2}$ (usually) including $\{SK_{U_1}, PK_{U_2}, id_{U_1}, id_{U_2}\}$ for user U_1 and U_2 are different from the inputs for generating $RK_{U_3 \to U_4}$ which usually includes $\{SK_{U_3}, PK_{U_4}, id_{U_3}, id_{U_4}\}$ for U_3 and U_4. This undoubtedly results in some inconvenience like re-encryption key management in the user or the RK generator side. Can we extract some common part(s) of those inputs for different RKs? In addition, inconvenience also exists in other aspects. For instance, the user may only trust her key generation server (key generator) but not the proxy; or a certain user U_1 may need to generate plenty of re-encryption keys for lots of other receivers $U_2, \ldots, U_k$, which may be a bottleneck for U_1 both in computation and in storage. All the above mentioned problems can be solved via such avenue: The key generator delegates his master secrete key MSK to the proxy without any privacy leakage about MSK. Then the proxy must have some special techniques to deal with the RK generation requirements between any pair of users. For the performance bottleneck of one specific user, since the RK generation workloads are all in the proxy server side, the possible busy user is then liberated from massive RK generation workloads.

In addition to email forwarding, there are a great many of other application scenarios for re-encryption in real life. To just name a few:

1) **Key revocation and key update.** Re-encryption is the mainstream technique and one of the most indispensable building blocks in schemes supporting key revocation (update). In such a typical scheme, after the key revocation procedure, the existing ciphertexts must be updated accordingly, involving tens of millions of existing ciphertexts. Without re-encryption, this ciphertext update procedure is definitely unbearable.

2) **Restricted law enforcement.** This is an example from [19] in which a law enforcement agency F wishes to scrutinize classified personal files of a set of suspect individuals G during a certain period of time. However, the legal court possessing all the keys cannot directly pass those keys to F, otherwise it will permanently obtain the privilege to infringe the privacy of these citizens. A plausible method is to let the court transform the ciphertext (with or without the help of a proxy) under a certain person's public key to the ciphertext under F's public key when F has already been granted a warrant. After the warrant loses its effect, F will lose its ability to probe into the classified files immediately.

3) **Fine-grained access control.** Suppose an accountant Alice at Department A before was a checker at Department B within the same corporation. Her private key corresponding to her identity is not only associated with her name "Alice" but also with her job title, *e.g.*, "Alice||Accountant" or "Alice||Checker". Then re-encryption allows for fine-grained access control so that she is able to deal with some of her final stage work after she leaved Department B.

1.2 Our Contribution and Main Technique

Our first contribution is to formalize the notion of homomorphic universal re-encryptor for identity-based encryption (HURE-IBE for short). This primitive is very useful in a scenario where the users only have limited computation or storage capability as we have explained above. Besides, the universality property features the advantage that a user can gain her re-encryption key quite easily by just issuing a re-encryption key query containing only two id's and the proxy does not need to deal with any public key or private key relevant information. This is due to the delegation of part of the master secret key s in the encrypted form under a fully homomorphic encryption (FHE) scheme.

Our scheme can not only provide solutions to practical issues, but also pave the way for follow-up research works.

Another contribution of this work is to put forward the first HURE-IBE scheme by combining the IBPRE scheme with an FHE scheme [4, 13]. In addition, although the proposed scheme in the construction part is instantiated via Boneh-Franklin IBE scheme, our construction is essentially generic. This means our general methodology can be applied to other IBPRE schemes, even other public-key proxy re-encryption schemes, subject to a condition that there must be a master secret key for the generation of all these user private keys.

To solve all those problems above mentioned, we have developed a novel technique to efficiently combine the possible IBPRE scheme and FHE scheme. A small shortcoming is that the introduction of FHE would decrease the efficiency of our scheme. But we argue that, on one hand, the scheme is more convenient than previous ones at the cost of a little more computation workloads; on the other hand, the state of the art FHE scheme is fast enough to bear the extra cost [23].

1.3 Related Works

Mambo *et al.* [21] first found the usefulness of re-encryption and suggested to use proxy cryptosystem to replace the trivial decrypt-and-encrypt method for sake of efficiency. Then proxy re-encryption and re-signature were conceptualized from a primitive named as atomic proxy function, which was coined by Blaze *et al.* [1]. Due to its wide range of application fields, proxy re-encryption has received much attention after its birth. For instance, since the earlier scheme for re-encryption proposed by Blaze *et al.* is inherently bidirectional, there are quite a number of works have focused on unidirectional schemes [19, 20]. For the security of the encryption schemes, chosen ciphertext attack (CCA) security is very important. Therefore there are also several works on CCA-secure PRE schemes [5, 16]. In addition, research on proxy signature schemes [18, 25] is also very active in cryptographic community.

Due to the significance, powerfulness and convenience of more advanced encryption schemes like identity-based encryption, attributed-based encryption [11], and functional encryption [3], re-encryption has also been developed along this line [6, 9, 14, 22]. Moreover, in recent years, since program obfuscation has a fast progress in several aspects, *e.g.*, extremely powerful constructions like indistinguishability obfuscation (iO) [10] has been developed, using obfuscation techniques to enhance re-encryption schemes is also a promising field. Besides those program obfuscation constructions for general programs/functions, special constructions also find their places due to their high efficiency. So the research field on re-encryption obfuscation is also very active. The first re-encryption obfuscation scheme is due to Hohenberger *et al.* [17], who had also proposed a new security definition framework for average-case secure obfuscation in order to bypass the limitations of obfuscating deterministic circuits. Then Hada [15] proposed obfuscation scheme for encrypted signatures. Chandran *et al.* [7] introduced collusion-resistant obfuscation to construct functional re-encryption scheme supporting function F's with a polynomial-size domain.

2 Preliminaries

Definition 1 (Identity-based proxy re-encryption). *An identity-based proxy re-encryption (IBPRE) scheme* Π_{RE} *consists of six probabilistic polynomial time (PPT) algorithms:*

- **Setup**$(n) \rightarrow (params, MSK)$: On input a security parameter n, the algorithm outputs the public parameters $params$ and the master secret key MSK, which should be kept secret. This algorithm may also decide the maximum encryption level of the cryptosystem under some conditions. This algorithm is run by the trust authority.

- **KeyGen**$(params, MSK, id) \rightarrow SK_{id}$: On input an identity $id \in \{0,1\}^*$, the master secret key MSK, and the public parameters $params$, the algorithm outputs a user secret key (decryption key) SK_{id} corresponding to the identity id. This algorithm is run by the trust authority.

- **Enc**$(params, id, m) \rightarrow c_{id}$: On input a plaintext $m \in \mathcal{M}$, an identity id, and the public parameters $params$, the algorithm outputs a ciphertext c_{id} which corresponds to the plaintext m and the identity id. This algorithm is run by the users.

- **Dec**$(params, SK_{id}, c_{id}) = m$: On input a ciphertext c_{id} which is the encryption of the plaintext m and the identity id, a user secret key SK_{id}, and the public parameters $params$, the deterministic algorithm outputs m if decryption succeeds; otherwise, it outputs $\perp$. This algorithm is run by the users.

- **RKGen**$(params, SK_{id_1}, id_1, id_2) \rightarrow RK_{id_1 \rightarrow id_2}$: On input two id's id_1, id_2, a user secret key SK_{id}, and the public parameters $params$, the algorithm outputs a re-encryption key $RK_{id_1 \rightarrow id_2}$. This algorithm may be run by the users or the proxy (server).

- **ReEnc**$(params, RK_{id_1 \rightarrow id_2}, c_{id_1}) \rightarrow c_{id_2}$: On input a ciphertext c_{id_1} under identity id_1, a re-encryption key $RK_{id_1 \rightarrow id_2}$, and the public parameters $params$, the algorithm outputs a re-encrypted ciphertext c_{id_2} under identity id_2. This algorithm may be run by the users or the proxy (server).

Remark 1. *Our scheme is similar but not consistent with the traditional descriptions. Especially, we will sometimes omit* $params$ *and treat it as an implicit input.*

Definition 2 (Fully homomorphic encryption). *A homomorphic encryption scheme* Π_{HE} *consists of four PPT algorithms:*

- **KeyGen**$_{\text{HE}}(n) \rightarrow (PK_{\text{HE}}, SK_{\text{HE}})$: This is a randomized algorithm which takes as input a security parameter n, and outputs a public key PK_{HE} and a secret key SK_{HE}.

- **Enc**$_{\text{HE}}(PK_{\text{HE}}, m) \rightarrow \text{CT}_{\text{HE}}(m)$: This is a randomized algorithm which takes as input a public key PK_{HE}, and a message $m \in \mathcal{M}$. It returns a ciphertext $\text{CT}_{\text{HE}}(m)$ as its output.

- **Dec**$_{\text{HE}}(SK_{\text{HE}}, \text{CT}_{\text{HE}}(m)) = m$: This is a deterministic algorithm which takes as input a secret key SK_{HE}, and a ciphertext $\text{CT}_{\text{HE}}(m)$. It returns the corresponding plaintext m if the decryption succeeds, and $\perp$ otherwise.

- $\mathbf{Eval}_{\mathrm{HE}}(PK_{\mathrm{HE}}, \Delta(\cdot), \mathrm{CT}_{\mathrm{HE}}(m_1), \ldots, \mathrm{CT}_{\mathrm{HE}}(m_k)) \to \mathrm{CT}_{\mathrm{HE}}(\Delta(m_1, \ldots, m_k))$: This is a randomized algorithm which takes as input a public key PK_{HE}, a circuit $\Delta(\cdot)$, and a bunch of ciphertexts $\{\mathrm{CT}_{\mathrm{HE}}(m_1), \ldots, \mathrm{CT}_{\mathrm{HE}}(m_k)\}$. It outputs a ciphertext $\mathrm{CT}_{\mathrm{HE}}(\Delta(m_1, \ldots, m_k))$ which is the encryption of the circuit output on inputs $m_1, \ldots, m_k$.

Remark 2. *In this work, we use fully homomorphic encryption as a black box, just like the way in some verifiable computation schemes [8, 12] for general circuits. For verifiable computation schemes for special functions like polynomial function [26], (proxy) signature scheme other than FHE might play a more significant role. Besides, for ease of description, we will sometimes omit the FHE public key PK_{HE} in some of the algorithms and notations, e.g., a ciphertext $\boldsymbol{Enc}_{HE}(PK_{HE}, m)$ of the homomorphic encryption scheme is equivalent to $CT_{HE}(m)$ in this work.*

Definition 3 (Bilinear maps)**.** *Let $\mathcal{G}$ be an algorithm which takes as input a security parameter n and outputs a tuple $(\hat{e}, q, g, \mathbb{G} = \langle g \rangle, \mathbb{G}_T = \langle \hat{e}(g, g) \rangle)$ where q is a large prime numbers, $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups of order q. A bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ has the following properties:*

1) **Bilinearity**: $\forall u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.

2) **Non-degeneracy**: If g generates $\mathbb{G}$, then $\hat{e}(g, g)$ generates $\mathbb{G}_T$.

3) **Efficiency**: Group operations in $\mathbb{G}$ and the bilinear map $\hat{e}$ are both computable in polynomial time.

Assumption 1 (Decisional Bilinear Diffie-Hellman (DBDH))**.** *The DBDH assumption says that the following two tuples are computationally indistinguishable.*

$$\{g, g^a, g^b, g^c, T = \hat{e}(g, g)^{abc}\} \overset{c}{\equiv} \{g, g^a, g^b, g^c, T \overset{\$}{\leftarrow} \mathbb{G}_T\}$$

3 IBPRE Constructions

3.1 Our Main IBPRE Scheme

Our main improvement on the traditional IBPRE scheme is to remove the possible burdensome RK generation workloads to the proxy whose role has usually been played by powerful cloud servers, and further cut down the total workloads. The main idea behind our scheme is that we use fully homomorphic encryption scheme to protect the master secret s and at the meantime to permit the required computations through the evaluation algorithm $\mathbf{Eval}_{\mathrm{HE}}$ of the FHE scheme. The construction methodology of ours is similar to the general verifiable computation protocol from Yao's Garbled Circuit and FHE proposed by Gennaro *et al.* [12]. More directly speaking, we use FHE to ensure the privacy and reusability of the master secret key. Then the proxy is able to securely and privately execute the RK generation procedures to respond to the RK generation requirements from the users without any privacy infringement of the master secret key.

We now provide a formal description of our scheme. Note that the common part of four algorithms are very similar to a common Boneh-Franklin IBE scheme except that there is an FHE system embedded in ours.

- $\mathbf{Setup}(n) \to (params, MSK)$: The **Setup** algorithm generates a bilinear map system $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, where $\mathbb{G} = \langle g \rangle$ and $\mathbb{G}_T = \langle \hat{e}(g, g) \rangle$ are both cyclic groups of order q, which is a large prime implicitly decided by the security parameter n. The algorithm also chooses two hash functions $H_1 : \{0,1\}^* \to \mathbb{G}$, $H_2 : \{0,1\}^* \to \mathbb{G}$, as well as a fully homomorphic encryption scheme $\Pi_{\mathrm{HE}} = \{\mathbf{KeyGen}_{\mathrm{HE}}, \mathbf{Enc}_{\mathrm{HE}}, \mathbf{Dec}_{\mathrm{HE}}, \mathbf{Eval}_{\mathrm{HE}}\}$ with a pair of FHE keys $(PK_{\mathrm{HE}}, SK_{\mathrm{HE}})$. In addition, the algorithm chooses a secret number s uniformly at random from $\mathbb{Z}_q^*$ as a part of the master secret key MSK, namely $s \overset{\$}{\leftarrow} \mathbb{Z}_q^*$. The public parameters are

$$params = \left(\hat{e}, \mathbb{G}, \mathbb{G}_T, g, g^s, H_1, H_2, \Pi_{\mathrm{HE}}, PK_{\mathrm{HE}}\right)$$

and the master secret key is $MSK = (s, SK_{\mathrm{HE}})$. Besides, $\mathrm{CT}_{\mathrm{HE}}(s)$ and SK_{HE} will be sent to the proxy and the users, respectively.

- $\mathbf{KeyGen}(params, MSK, id) \to SK_{id}$: The **KeyGen** algorithm outputs a user secret key $SK_{id} = H_1(id)^s$ for the input identity $id \in \{0,1\}^*$.

- $\mathbf{Enc}(params, id, m) \to c_{id}$: In order to encrypt a message m under an identity id, the **Enc** algorithm selects $r \overset{\$}{\leftarrow} \mathbb{Z}_q^*$ uniformly at random and outputs the ciphertext

$$c_{id} = (C_1, C_2) = (g^r, m \cdot \hat{e}(g^s, H_1(id))^r)$$

- $\mathbf{RKGen}(params, \mathrm{CT}_{\mathrm{HE}}(s), id_1, id_2) \to RK_{id_1 \to id_2}$: To generate a RK from id_1 to id_2, the **RKGen** algorithm must generate three parts. It first homomorphically runs the FHE evaluation algorithm to conduct the following evaluation on circuit Δ_1 and the three FHE ciphertexts:

$$\begin{aligned}
&\mathbf{Eval}_{\mathrm{HE}}\Big(\Delta_1, \mathrm{CT}_{\mathrm{HE}}(id_1), \mathrm{CT}_{\mathrm{HE}}(s), \mathrm{CT}_{\mathrm{HE}}(id_2)\Big) \\
=&\mathbf{Enc}_{\mathrm{HE}}\Big(PK_{\mathrm{HE}}, SK_{id_1}^{-1} \cdot H_2(\mathbf{str}(K_{id_1 id_2})||id_1 \to id_2)\Big) \\
=&\mathbf{Enc}_{\mathrm{HE}}\Big(PK_{\mathrm{HE}}, SK_{id_1}^{-1} \cdot \\
&\qquad H_2(\underbrace{\mathbf{str}(\hat{e}(H_1(id_1), H_2(id_2))^s)||id_1 \to id_2}_{\mathrm{Const}_{id_1 \to id_2}})\Big) \\
=&\mathrm{CT}_{\mathrm{HE}}(SK_{id_1}^{-1} \cdot H_2(\underbrace{\mathrm{Const}_{id_1 \to id_2}}_{X}))
\end{aligned}$$

Here some notations in the above formula should be more detailedly explained. $\mathbf{str}(\cdot)$ is a function that outputs the bit-string representation of its input. And below we will use a simpler notation X to denote $\mathrm{Const}_{id_1 \to id_2}$. Besides, we will illustrate circuit Δ_1 as the following 3-ary function.

$$\begin{aligned}\Delta_1(x_1, x_2, x_3) = (H_1(x_1)^{x_2})^{-1} \cdot \\ H_2(\mathbf{str}(\hat{e}(H_1(x_1), H_2(x_3))^{x_2} \| x_1 \to x_3))\end{aligned}$$

That is to say, let the three inputs x_1, x_2, x_3 be id_1, s, id_2 respectively, we have

$$\begin{aligned}\Delta_1(id_1, s, id_2) = (H_1(id_1)^s)^{-1} \cdot \\ H_2(\mathbf{str}(\hat{e}(H_1(id_1), H_2(id_2))^s \| id_1 \to id_2)) \\ = SK_{id_1}^{-1} \cdot H_2(\mathrm{Const}_{id_1 \to id_2}) \\ = SK_{id_1}^{-1} \cdot H_2(X)\end{aligned}$$

The other two parts can be generated by simple IBE encryption operations on X and then the resulting two parts of the ciphertext are encrypted by the FHE algorithm $\mathbf{Enc}_{\mathrm{HE}}$. So the final re-encryption key from id_1 to id_2 is

$$RK_{id_1 \to id_2} = \big(\mathrm{CT}_{\mathrm{HE}}(RK_1), \mathrm{CT}_{\mathrm{HE}}(RK_2), \mathrm{CT}_{\mathrm{HE}}(RK_3)\big)$$

where we have,

$$\begin{cases} \mathrm{CT}_{\mathrm{HE}}(RK_1) = \mathrm{CT}_{\mathrm{HE}}(g^{r'}) \\ \mathrm{CT}_{\mathrm{HE}}(RK_2) = \mathrm{CT}_{\mathrm{HE}}(X \cdot \hat{e}(g^s, H_1(id_2))^{r'}) \\ \mathrm{CT}_{\mathrm{HE}}(RK_3) = \mathrm{CT}_{\mathrm{HE}}(SK_{id_1}^{-1} \cdot H_2(X)) \end{cases}$$

Besides, we also denote the first two components of the RK, $\mathrm{CT}_{\mathrm{HE}}(RK_1)$ and $\mathrm{CT}_{\mathrm{HE}}(RK_2)$, as $\mathrm{CT}_{\mathrm{HE}}(\mathbf{Enc}(params, id_2, X))$ because they are actually an IBE ciphertext of X and randomness r' regardless of the FHE encryption layer.

- $\mathbf{ReEnc}(params, RK_{id_1 \to id_2}, c_{id_1}) \to c_{id_2}$: To produce a re-encrypted ciphertext for the input ciphertext c_{id_1} with the form of

$$c_{id_1} = (C_1, C_2) = (g^r, m \cdot \hat{e}(g^s, H_1(id_1))^r)$$

the **ReEnc** algorithm must generate four parts. It first runs the FHE evaluation algorithm $\mathbf{Eval}_{\mathrm{HE}}$ on the homomorphic encryption of c_{id_1}, and the third part of the re-encryption key $RK_{id_1 \to id_2}$ to generate the second part of the re-encrypted ciphertext, namely,

$$\begin{aligned}&\mathbf{Eval}_{\mathrm{HE}}\Big(\Delta_2, \mathrm{CT}_{\mathrm{HE}}(C_2), \mathrm{CT}_{\mathrm{HE}}(C_1), \mathrm{CT}_{\mathrm{HE}}(RK_3)\Big) \\ =&\mathbf{Eval}_{\mathrm{HE}}\Big(\Delta_2, \mathrm{CT}_{\mathrm{HE}}(m \cdot \hat{e}(g^s, H_1(id_1))^r), \mathrm{CT}_{\mathrm{HE}}(g^r), \\ &\mathrm{CT}_{\mathrm{HE}}(SK_{id_1}^{-1} \cdot H_2(X))\Big) \\ =&\mathbf{Enc}_{\mathrm{HE}}\Big(PK_{\mathrm{HE}}, m \cdot \hat{e}(g, H_2(X))^r\Big) \\ =&\mathrm{CT}_{\mathrm{HE}}\Big(m \cdot \hat{e}(g, H_2(X))^r\Big)\end{aligned}$$

The 3-ary circuit Δ_2 does the following computation:

$$\Delta_2(x_1, x_2, x_3) = x_1 \cdot \hat{e}(x_2, x_3)$$

Therefore, when the three inputs x_1, x_2, x_3 are C_2, C_1, RK_3 respectively

$$\begin{aligned}&\Delta_2(C_2, C_1, RK_3) \\ =&m \cdot \hat{e}(g^s, H_1(id_1))^r \cdot \hat{e}(g^r, SK_{id_1}^{-1} \cdot H_2(X)) \\ =&m \cdot \hat{e}(g, H_1(id_1))^{rs} \cdot \hat{e}(g^r, SK_{id_1}^{-1}) \cdot \hat{e}(g^r, H_2(X)) \\ =&m \cdot \hat{e}(g, H_2(X))^r\end{aligned}$$

For the other three parts, they can be easily obtained - actually there is no need to do further computation. The final re-encrypted ciphertext $c_{id_2}^{\mathrm{RE}}$ is,

$$\begin{aligned}c_{id_2}^{\mathrm{RE}} =&(C_1, C_2, C_3, C_4) \\ =&\big(g^r, \mathrm{CT}_{\mathrm{HE}}(m \cdot \hat{e}(g, H_2(X))^r), \mathrm{CT}_{\mathrm{HE}}(RK_1), \\ &\mathrm{CT}_{\mathrm{HE}}(RK_2)\big)\end{aligned}$$

- $\mathbf{Dec}(params, SK_{id}, c_{id}) = m$: Decryption is categorized into two types according to the ciphertext type.
  - Condition 1: If the input ciphertext is a normal IBE ciphertetx with the form of $c_{id} = (C_1, C_2) = (g^r, m \cdot \hat{e}(g^s, H_1(id))^r)$, the decryption algorithm will just do the Boneh-Franklin IBE decryption calculation, namely,

$$m = \mathbf{Dec}_{\mathrm{IBE}}(C_1, C_2) = C_2 / \hat{e}(C_1, SK_{id})$$

  - Condition 2: If the input ciphertext is a re-encrypted ciphertext with the form of $c_{id}^{\mathrm{RE}} = (C_1, C_2, C_3, C_4) = \big(g^r, \mathrm{CT}_{\mathrm{HE}}(m \cdot \hat{e}(g, H_2(X))^r), \mathrm{CT}_{\mathrm{HE}}(RK_1), \mathrm{CT}_{\mathrm{HE}}(RK_2)\big)$, the decryption algorithm will in addition invoke the FHE decryption algorithm to obtain the plaintext encrypted under PK_{HE}. More specifically, it first decrypts $\mathrm{CT}_{\mathrm{HE}}(RK_1)$, $\mathrm{CT}_{\mathrm{HE}}(RK_2)$ to obtain X. And then it uses X like a "private key" to decrypt the former two parts in the type-II ciphertext.

$$\begin{aligned}X &= \mathbf{Dec}_{\mathrm{IBE}}(\mathbf{Dec}_{\mathrm{HE}}(C_3), \mathbf{Dec}_{\mathrm{HE}}(C_4)) \\ m &= \mathbf{Dec}_{\mathrm{HE}}(C_2) / \hat{e}(C_1, H_2(X))\end{aligned}$$

3.2 Two Variants to Resist Possible FHE Key Leakage

Note that in our settings, we make a moderate assumption that a legal user will not intentionally or unintentionally leak the private homomorphic decryption key SK_{HE} to any parties, especially to the proxy. This assumption is indeed realistic since in real life the role of a proxy server is frequently played by the cloud computing units of those Internet giants like Amazon's AWS or Microsoft's Azure whose commercial reputations are of vital importance. Nevertheless, we still provide two countermeasures to cope with this issue, which give rise to two variants of our main construction. As the main aim of our scheme is to largely reduce the workload in the user side, the two variants will lose some advantages as tradeoffs compared with the main construction in a bid to achieve a higher security goal. Bellow we briefly illustrate the two countermeasures as well as their pros and cons in comparison to the scheme in Section 3.1.

- **Variant I. (Deprive of the full decryption power from a single user)**

  The most direct avenue to deal with possible private homomorphic decryption key leakage is to make a shift in the beginning that the trust authority does not share SK_{HE} to every user. Instead, it has two alternative strategies. The first one is to conduct the outermost homomorphic decryption procedures completely by itself. The main merit is that it has actually solved the FHE private key leakage problem. Nevertheless, potential decryption bottleneck in the authority side might be introduced. The second strategy is to share SK_{HE} to a threshold number of users, following the similar vein of the first strategy, namely, incapacitating a single user for her decryption ability. The shortcomings for this method is that the collaboration of more than one user is a must for conducting a decryption operation.

- **Variant II. (Introduce distinct FHE key pairs to the system for different users)**

  An alternative way to deal with possible private homomorphic decryption key leakage is decentralization. More specifically, the trust authority does not control over the homomorphic encryption system any more. On the contrary, a different homomorphic key pair per user will be used in the **RKGen** procedure to replace the global homomorphic encryption-decryption key pair. Accordingly, other related procedures like **ReEnc** should also be modified. In this way, once an FHE key pair for user U_i is compromised, the only victim is just user U_i, which rules out deliberate FHE key leakage. However, we point out that this change will also lead to inefficiency (due to the involvements of more FHE key pairs) and inconvenience.

4 Security and Efficiency

We first provide mathematical deductions for the correctness of our scheme.

Correctness. The correctness is guaranteed by the following formulas.
Condition 1:

$$C_2/\hat{e}(C_1, SK_{id}) = \frac{m \cdot \hat{e}(g^s, H_1(id))^r}{\hat{e}(g^r, H_1(id)^s)} = m$$

Condition 2:

$$\begin{aligned}
&\mathbf{Dec}_{\mathrm{IBE}}(\mathbf{Dec}_{\mathrm{HE}}(C_3), \mathbf{Dec}_{\mathrm{HE}}(C_4))\\
=&\mathbf{Dec}_{\mathrm{IBE}}(RK_1, RK_2)\\
=&RK_2/\hat{e}(RK_1, SK_{id})\\
=&X
\end{aligned}$$

$$\mathbf{Dec}_{\mathrm{HE}}(C_2)/\hat{e}(C_1, H_2(X)) = \frac{m \cdot \hat{e}(g, H_2(X))^r}{\hat{e}(g^r, H_2(X))} = m$$

Security. Then it goes to the security proof. Actually, our proof of security is mainly divided into two parts. The first part is to prove that the bare scheme Π'_{HURE} without the fully homomorphic encryption scheme Π_{HE} is secure and then prove that the full scheme is secure. In fact, due to the extensive research on FHE, we just need to prove the security of the bare scheme.

Proof. The proof is constructed by contradiction. Suppose there is an adversary $\mathcal{A}$ who has a non-negligible advantage in attacking the scheme Π'_{HURE}, then we can construct another adversary $\mathcal{A}'$ to succeed in attacking the DBDH problem with a non-negligible advantage.

Suppose $\mathcal{A}'$ receives a tuple $\langle g, g^a, g^b, g^c, T\rangle$ from the challenger $\mathcal{C}$, which may be a DBDH tuple with $T = \hat{e}(g,g)^{abc}$ or a random tuple with $T \xleftarrow{\$} \mathbb{G}_T$. To take advantage of $\mathcal{A}$, adversary $\mathcal{A}'$ must prepare parameters and respond to the queries by $\mathcal{A}$. We then illustrate how $\mathcal{A}'$ finishes these steps.

Setup: Adversary $\mathcal{A}'$ first establishes the system parameters as $params = (\hat{e}, \mathbb{G} = \langle g\rangle, \mathbb{G}_T = \langle \hat{e}(g,g)\rangle, g, g^a, H_1, H_2)$. There are two points worth mentioning. First, there is no need to include the homomorphic encryption scheme as well as the public key PK_{HE} of this scheme into $params$, since here we just consider the security of the bare scheme as stated above. Second, although the adversary does not know a, she can still use g^a to replace g^s of the original scheme, since g^a is already in the tuple. Beside, the adversary $\mathcal{A}'$ also maintains a table $\mathcal{T}$ to record the responses.

Simulate hash queries: To simulate $H_1 : \{0,1\}^* \to \mathbb{G}$, the adversary $\mathcal{A}'$ responds as follow. On an input identity id, if $id = id^*$, which is the challenge identity, the response is $h \leftarrow (g^c)^z$, where $z \xleftarrow{\$} \mathbb{Z}_q^*$; otherwise, the response is just $h \leftarrow g^z$, where $z \xleftarrow{\$} \mathbb{Z}_q^*$. To simulate $H_2 : \{0,1\}^* \to \mathbb{G}$, the adversary $\mathcal{A}'$ just returns a random element in $\mathbb{G}$. After each query, the corresponding results *e.g.*, $\{id_i, h_i, z_i\}$ must be included into the table $\mathcal{T}$. Besides, we underline that whenever an id query comes, the adversary $\mathcal{A}'$ must first look up the table $\mathcal{T}$ to find out if id has already been queried. If so, $\mathcal{A}'$ should locate the existing tuple at the table and return the corresponding content. This is also the case for key queries.

Simulate key queries: On a user secret key query for id_i, the adversary $\mathcal{A}'$ first generates the corresponding tuple $\{id_i, h_i, z_i\}$ as stated above. Subsequently, it returns $SK_{id_i} = (g^a)^{z_i}$ as the response, which will then be added up to the tuple in table $\mathcal{T}$. For a re-encryption key query $id_1 \rightarrow id_2$, if this query will lead to a trivial decryption of the challenge ciphertext, the adversary $\mathcal{A}'$ returns

$$RK_{id_1 \rightarrow id_2} = \big((g^b)^r, T^{rz_2} \cdot X, g^x\big)$$

where $r, x \overset{\$}{\leftarrow} \mathbb{Z}_q^*$ and X is computed as stated in the scheme - actually, it can be selected uniformly at random. Note that this RK is not correctly formed, but the adversary $\mathcal{A}$ can not detect this. If a RK query will not lead to a trivial decryption of the challenge ciphertext, the adversary $\mathcal{A}'$ returns

$$RK_{id_1 \rightarrow id_2} = \big(g^r, X \cdot \hat{e}(g^a, H_1(id_2))^r, (g^a)^{-z_1} \cdot H_2(X)\big)$$

another form of which is $\big(\mathbf{Enc}(id_2, X), SK_{id_1}^{-1} \cdot H_2(X)\big)$.

Formalize challenge ciphertext: In this step, the adversary $\mathcal{A}'$ receives two equal-length message m_0, m_1 as inputs from $\mathcal{A}$. To formalize the challenge ciphertext c^*, the adversary $\mathcal{A}'$ first evaluates $\{id^*, h^*, z^*\}$ if the hash query of id^* has not been issued previously. Then $\mathcal{A}'$ flips a random coin $f \in \{0, 1\}$ and returns $c^* = (g^b, T^{z^*} \cdot m_f)$.

Note that after the generation of challenge ciphertext c^*, additional queries are also permitted conditioned on two requirements: First, the total number of queries is bounded by a fixed value which we do not explicitly state here. And second, any query which will lead to a trivial decryption for c^* is prohibited.

We argue that although some components are not correctly formed, the adversary $\mathcal{A}$ can not detect this, meaning that the adversary $\mathcal{A}$ can not distinguish the simulation by $\mathcal{A}'$ without the secret s (but with g^a) from a real execution by a real challenge $\mathcal{C}$ who possesses the secret s. Therefore, from her point of view, these two procedures are computationally identical. For ease of description, just take the two kinds of RK's as an example. Since r is a random element, it is easy to see that $(g^b)^r \overset{c}{\equiv} g^r$ and it is the same for either $T^{rz_2} \cdot X$ and $X \cdot \hat{e}(g^a, H_1(id_2))^r$, as well as g^x and $(g^a)^{-z_1} \cdot H_2(X)$.

At the end of all the interactions between adversary $\mathcal{A}'$ and $\mathcal{A}$, $\mathcal{A}$ must return her guess to $\mathcal{A}'$. The latter will make a choice on whether $T = \hat{e}(g, g)^{abc}$ or $T \overset{\$}{\leftarrow} \mathbb{G}_T$ according to the guess bit f' returned by $\mathcal{A}$. If $f' = f$, namely, adversary $\mathcal{A}$ succeeds in attacking the bare scheme, then $\mathcal{A}'$ will return 1, indicating that $T = \hat{e}(g, g)^{abc}$ is belonging to a DBDH tuple; otherwise, $\mathcal{A}'$ will return 0, indicating that $T \overset{\$}{\leftarrow} \mathbb{G}_T$ is belonging to a random tuple.

To see how it works, we will discuss according to the value of T. When $T = \hat{e}(g, g)^{abc}$, the challenge ciphertext has the following form:

$$\begin{aligned} c^* &= (g^b, T^{z^*} \cdot m_f) = (g^b, \hat{e}(g, g)^{abcz^*} \cdot m_f) \\ &= (g^b, \hat{e}(g^a, g^{cz^*})^b \cdot m_f) \\ &= (g^b, m_f \cdot \hat{e}(g^a, H_1(id^*))^b) \end{aligned}$$

In other word, it is a correctly formed ciphertext. So in this game, the adversary $\mathcal{A}$ has a non-negligible advantage ϵ in distinguishing whether c^* is the encryption of m_0 or the encryption of m_1.

When $T \overset{\$}{\leftarrow} \mathbb{G}_T$, the challenge ciphertext has the following form (r is a random element selected from $\mathbb{Z}_q^*$, which makes rz^* a random element in $\mathbb{Z}_q^*$. And R is thus a random element selected from $\mathbb{G}_T$):

$$\begin{aligned} c^* &= (g^b, T^{z^*} \cdot m_f) = (g^b, \hat{e}(g, g)^{rz^*} \cdot m_f) \\ &= (g^b, m_f \cdot R) \end{aligned}$$

Since R is a random element in $\mathbb{G}_T$, the probability for adversary $\mathcal{A}'$ to successfully guess the bit f is just $\frac{1}{2}$, *i.e.*, the advantage for her is 0. So in terms of whether adversary $\mathcal{A}$ succeeds in the Π'_{HURE} game, adversary $\mathcal{A}'$ can also have a non-negligible advantage in distinguishing a DBDH tuple from a random tuple. □

Efficiency. The main advantage of our scheme is the universality property mentioned before, which has never been achieved in any other PRE schemes. To this end, we sacrifice some efficiency as the tradeoffs. Nevertheless, our scheme has gained advantages in communication complexity aspects since the user-specific private information is no longer needed to send.

Below we will provide the readers with a simplified analysis of the efficiency, namely we roughly divide various operations in IBPRE schemes into three categories according to an approximate and empirical evaluation criterion.

1) T_1: Hash operations, including hash operations $H_1(\cdot)$ and $H_2(\cdot)$.

2) T_2: Group operations, including operations in both $\mathbb{G}$ and $\mathbb{G}_T$ as well as FHE operations.

3) T_3: Bilinear pairing operations.

Then the detailed comparisons of computation complexity between our scheme and the most relevant scheme [14] are described in Table 1.

The main reason why we do not take into consideration the efficiency of **Setup** and **KeyGen** is that these two algorithms are run by the trust authority while in reality we care more about the efficiency of the users as well as the proxy. Besides, they are not the core factors for a cryptosystem since the number of running time of them are far less than that of the left four algorithms. For the fourth row of Table 1, $\{0\ (3T_2)\}$ means that in our scheme, the

Table 1: Comparison of computation complexity

	Our scheme	Scheme of [14]
Enc	$T_1 + 3T_2 + T_3$	$T_1 + 3T_2 + T_3$
RKGen	$T_1 + 8T_2 + T_3$	$2T_1 + 5T_2 + T_3$
ReEnc	$0\ (3T_2)$	$T_2 + T_3$
Dec$^{(1)}$	$T_2 + T_3$	$T_2 + T_3$
Dec$^{(2)}$	$5T_2 + 2T_3$	$2T_2 + 2T_3$

user does not need to do any computation and therefore $3T_2$ is merely for the proxy while $\{T_2 + T_3\}$ is for the user in their scheme. For the fifth and sixth row, **Dec**$^{(1)}$ means the normal IBE decryption and **Dec**$^{(2)}$ means the decryption of a re-encrypted ciphertext. From Table 1 we can see that our scheme only introduces bearable additional computation workload to some of the algorithms like **Dec**$^{(2)}$ while ours performs better in **ReEnc**. Besides, the special property like universality can only be achieved by our scheme. So in general, ours is of significant usefulness in both theoretical research on re-encryption and some real-life applications.

5 Conclusion and Future Work

In this work, we have introduced a new type of identity-based proxy re-encryption scheme, which is different from as well as superior over the former schemes in some aspects. Specifically, Our scheme enjoys a good feature that there is only very little resource needed in the user side. Besides, potential bottlenecks like re-encryption management are also avoided. However, our scheme also has some limitations, *e.g.*, the introduction of homomorphic encryption decreases the efficiency of the scheme and the scheme is not suitable for multi-hop re-encryptions. We will continue working on enhancing our scheme and constructing more powerful ones.

Acknowledgement

The work described in this paper is supported by the Opening Project of Sichuan Province University Key Laboratory of Bridge Non-destruction Detecting and Engineering Computing (No. 2015QYJ04); Science Founding of Artificial Intelligence Key Laboratory of Sichuan Province (No. 2014RYJ06); Sichuan Provincial Key Laboratory of University Enterprise Information and Network Measurement and Control Technology (No. 2014WYJ03); The Scientific Research Fund Project of Sichuan University of Science & Engineering (No. 2013KY02).

References

[1] M. Blaze, G. Bleumer and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Advances in Cryptology (Eurocrypt'98)*, pp. 127–144, Springer, 1998.

[2] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586–615, 2003.

[3] D. Boneh, A. Sahai and B. Waters, "Functional encryption: Definitions and challenges," in *Theory of Cryptography*, pp. 253–273, Springer, 2011.

[4] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 831–871, 2014.

[5] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pp. 185–194, 2007.

[6] N. Chandran, M. Chase, F. H. Liu, R. Nishimaki and K. Xagawa, "Re-encryption, functional re-encryption, and multi-hop re-encryption: A framework for achieving obfuscation-based security and instantiations from lattices," in *Public-Key Cryptography (Pkr'14)*, pp. 95–112, Springer, 2014.

[7] N. Chandran, M. Chase and V. Vaikuntanathan, "Functional re-encryption and collusion-resistant obfuscation," in *Theory of Cryptography*, pp. 404–421, Springer, 2012.

[8] K. M. Chung, Y. Kalai and S. Vadhan, "Improved delegation of computation using fully homomorphic encryption," in *Advances in Cryptology (Crypto'10)*, pp. 483–501, Springer, 2010.

[9] P. S. Chung, C. W. Liu and M. S. Hwang, "A study of attribute-based proxy re-encryption scheme in cloud environments," *International Journal of Network Security*, vol. 16, no. 1, pp. 1–13, 2014.

[10] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai and B. Waters, "Candidate indistinguishability obfuscation and functional encryption for all circuits," in *IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 40–49, 2013.

[11] S. Garg, C. Gentry, S. Halevi, A. Sahai and B. Waters, "Attribute-based encryption for circuits from multilinear maps," in *Advances in Cryptology (Crypto'13)*, pp. 479–499, Springer, 2013.

[12] R. Gennaro, C. Gentry and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Advances in Cryptology (Crypto'10)*, pp. 465–482, Springer, 2010.

[13] C. Gentry et al., "Fully homomorphic encryption using ideal lattices," in *Annual Symposium on the Theory of Computing*, vol. 9, pp. 169–178, 2009.

[14] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *Applied Cryptography and Network Security*, pp. 288–306, Springer, 2007.

[15] S. Hada, "Secure obfuscation for encrypted signatures," in *Advances in Cryptology (Eurocrypt'10)*, pp. 92–112, Springer, 2010.

[16] G. Hanaoka, Y. Kawai, N. Kunihiro, T. Matsuda, J. Weng, R. Zhang and Y. Zhao. "Generic

construction of chosen ciphertext secure proxy re-encryption," in *Topics in Cryptology (Ct-rsa'12)*, pp. 349–364, Springer, 2012.

[17] S. Hohenberger, G. N. Rothblum, V. Vaikuntanathan, et al., "Securely obfuscating re-encryption," in *Theory of Cryptography*, pp. 233–252. Springer, 2007.

[18] M. S. Hwang, C. C. Lee and S. F. Tzeng, "A new proxy signature scheme for a specified group of verifiers," *Information Sciences*, vol. 227, pp. 102–115, 2013.

[19] A. A. Ivan and Y. Dodis, "Proxy cryptography revisited," in *Network and Distributed System Security Symposium*, 2003.

[20] B. Libert and D. Vergnaud, "Unidirectional chosen-ciphertext secure proxy re-encryption," in *Public Key Cryptography (Pkc'08)*, pp. 360–379, Springer, 2008.

[21] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decrypt ciphertexts," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 80, no. 1, pp. 54–63, 1997.

[22] T. Matsuo, "Proxy re-encryption systems for identity-based encryption," in *Pairing-Based Cryptography (Pairing'07)*, pp. 247–267, Springer, 2007.

[23] M. Naehrig, K. Lauter and V. Vaikuntanathan, "Can homomorphic encryption be practical?," in *Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop*, pp. 113–124, 2011.

[24] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology*, pp. 47–53, Springer, 1985.

[25] F. Wang, C. C. Chang, C. Lin and S. C. Chang, "Secure and efficient identity-based proxy multi-signature using cubic residues," *International Journal of Network Security*, vol. 18, no. 1, pp. 90–98, 2016.

[26] J. Ye, H. Zhang and C. Fu, "Verifiable delegation of polynomials," *International Journal of Network Security*, vol. 18, no. 2, pp. 283–290, 2016.

Liang Liu received his B.S. degree in Computer Science and M.S. degree in Cryptography, both at School of Computer Science, Shaanxi Normal University. He is now a Ph.D. candidate in Cryptography at Xidian University. His research interests include theoretical cryptography, provable security and functional encryption.

Jun Ye received his B.S. degree in Applied Mathematics at Chongqing University and M.S. degree in Cryptography at Guilin University of Electronic Technology. He is now a Ph.D. candidate in Cryptography and Cloud Computing Security at Xidian University. His research interests include cryptography and information security.

An SVEIR Defending Model with Partial Immunization for Worms

Fangwei Wang[1,2], Hongfeng Gao[3], Yong Yang[4], and Changguang Wang[1,2]
(Corresponding author: Changguang Wang)

College of Information Technology, Hebei Normal University[1]
No. 20, South ErHuan Road, YuHua District, Shijiazhuang 050024, China
Shaanxi Key Laboratory of Network and System Security, Xidian University[2]
No. 2, TaiBai South Road, YanTa District, Xi'an 710071, China
Network and Information Center, Guizhou University[3]
No. 242, Huaxi Street, HuaXi District, Guiyang 550025, China
Network and Information Center, Yunnan University[4]
No. 2, Cuihu North Road, WuHua District, Kunming 650091, China
(Email: wangcg@hebtu.edu.cn)

(Received Oct. 28, 2015; revised and accepted Jan. 23, 2016)

Abstract

Internet worms can propagate across networks horrendously, reduce network security remarkably, and cause economic losses heavily. How to quickly eliminate the Internet worms using partial immunization becomes a big issue for sustaining Internet infrastructure smoothly. This paper addresses this issue by presenting a novel worm attack model through incorporating a saturated incidence rate and a partial immunization rate, named *SVEIR* model. Using the basic reproduction number, we derive the global stability of the infection-free equilibrium and local stability of the unique endemic equilibrium. Numerical methods are employed to solve and simulate the developed system and also verify the proposed *SVEIR* model. Simulation results show that the partial immunization is highly effective for eliminating worms.

Keywords: Internet worm, partial immunization, propagation model, saturated incidence, stability

1 Introduction

Internet worms are malicious codes which can replicate themselves and propagate across the Internet. Code red worm, Slammer worm, Blaster worm, Witty worm, and Conficker worm are a few examples of Internet worms, which have caused heavy economic losses and tremendous social panic. Especially, with the advent and development of the Internet of Things (IoT), the threat of Internet worms will become increasingly serious for network security. Combating worms effectively is an urgent task confronted with defenders. Based on the similarity between a malicious worm and a biological virus, a few mathematical models representing worm propagation have been presented to depict the propagation of worms in the past decade years [4, 9, 13]. Appropriate mathematical models can provide a qualitative assessment for worms' attack. Many numerous models and tools are proposed to address the dynamic attacking behaviour of worms and effectively counterattack them in different conditions, e.g., time delay [15, 18], quarantine [19, 20], antivirus software [21], etc. All the previous models are based on the *SIR* classical epidemic model [7]. The *SIR* has some drawbacks because it assumes that a susceptible host becomes infectious immediately after contact with an infected one. Actually, many worms own an exposed period during which susceptible hosts are infected but not yet contagious. To overcome this drawback, a new model, named as *SEIR* model [1], is introduced. An exposed class is added into the *SEIR* model.

Immunization is one of the commonly used methods for controlling and eliminating worms propagation [5, 6]. However, these models assumed that the vaccine hosts obtained the immunization fully. This is not consistent with the reality. For worms' horrendous propagation speed, users or network administrators can not immunize the whole host population in real networks. Thus, partial immunization as a fungible and feasible method for eliminating worms has been used for predicting and controlling infectious diseases [2, 12]. In many worm propagation models [5, 14], bilinear infection rate βSI is used, where, S and I denote the number of susceptible hosts and infectious hosts, respectively. The saturated infection rate $\frac{\beta SI}{1+\eta I}$ was firstly introduced by Capasso and Seior [3], where $\frac{\beta I}{1+\eta I}$

tends to a saturation level when I becomes large, $\frac{1}{1+\eta I}$ measures the inhibition effect from the behavioral change of susceptible hosts when their number increases or from the crowding effect of the infected hosts. The saturated infection rate $\frac{\beta SI}{1+\eta I}$ is more reasonable than the linear rate βSI. This is due to the fact that it takes the behavioral change and the effect of the infected hosts into consideration.

In this paper, we propose an *SVEIR* model based on the *SEIR* model. Contrary to existing models, the proposed *SVEIR* model is armed with the partial immunization and saturated infection. This paper will argue that *SVEIR* model is appropriate for measuring the effects of security countermeasures on worm propagation. Using the basic reproduction number, we derive global stability of the infection-free equilibrium and local stability of the unique endemic equilibrium.

The rest of this paper is organized as follows. Section 2 formulates the extended *SVEIR* model, which takes two important factors: Partial immunization and a saturated incidence rate, and obtains the basic reproduction number. Section 3 proves the stabilities of the equilibria. Section 4 covers the numerical analysis and the simulations. Section 5 summarizes the paper with some future directions.

2 A Mathematical Formulation for SVEIR Model

This section will examine the *SVEIR* model with a mathematical formulation. The *SVEIR* model extends the classical *SEIR* model through incorporating a saturated incidence rate and a partial immunization rate. The total host population N is divided into five groups and a host at any time t can potentially be in one of the following groups: Susceptible, vaccinated, exposed, infectious, recovered, with sizes denoted by S, V, E, I, R, respectively. The total number of population N at time t is given by $N(t) = S(t) + V(t) + E(t) + I(t) + R(t)$. The dynamic transition of the hosts is shown in Figure 1.

In Figure 1, Π is the constant recruitment rate of the host population, μ is the natural death rate of the population, and α is the death rate for worm attack of infectious hosts. Let β be the transmission rate of worm attack when susceptible hosts contact with infected ones. p is the fraction of recruited hosts which are vaccinated. γ is the rate at which vaccine wanes. The emergence of this scenario is due to worm variants. η is the parameter measuring the inhibitory effect. $\frac{\beta SI}{1+\eta I}$ is the saturated infection rate. ω is the rate at which exposed hosts become infectious, and δ is the recovered rate of infected hosts. The vaccinated hosts which contact infected ones before obtaining immunization have the infection probability with a transmission rate $\sigma\beta$ $(0 \le \sigma \le 1)$. $\sigma = 0$ means that the vaccinated hosts obtain the full immunization, which $\sigma = 1$ means that vaccine loses efficacy in work fully. Taking some real factors into account, we assume that the vaccinated hosts can obtain partial immunization, i.e. $0 < \sigma < 1$.

Based on the above assumptions, the *SVEIR* worm propagation model with partial immunization in the host population is described by the following system of differential equations:

$$\begin{cases} S'(t) = (1-p)\Pi - \frac{\beta SI}{1+\eta I} - \mu S + \gamma V, \\ V'(t) = p\Pi - \sigma\beta VI - (\mu+\gamma)V, \\ E'(t) = \frac{\beta SI}{1+\eta I} + \sigma\beta VI - (\mu+\omega)E, \\ I'(t) = \omega E - (\mu+\alpha+\delta)I, \\ R'(t) = \delta I - \mu R. \end{cases} \quad (1)$$

Since the state R does not appear explicitly in the first four equations in (1), the dynamics of (1) is the same as the following system:

$$\begin{cases} S'(t) = (1-p)\Pi - \frac{\beta SI}{1+\eta I} - \mu S + \gamma V, \\ V'(t) = p\Pi - \sigma\beta VI - (\mu+\gamma)V, \\ E'(t) = \frac{\beta SI}{1+\eta I} + \sigma\beta VI - (\mu+\omega)E, \\ I'(t) = \omega E - (\mu+\alpha+\delta)I. \end{cases} \quad (2)$$

Summing equations in (2), we obtain $(S+V+E+I)' = \Pi - \mu(S+V+E+I) - (\alpha+\delta)I \le \Pi - \mu(S+V+E+I)$. Then it follows that $\limsup_{t\to\infty}[S(t)+V(t)+E(t)+I(t)] \le \Pi/\mu$, thus the set

$$\Omega = \{(S,V,E,I) \in \mathcal{R}^4 : S+V+E+I \le \Pi/\mu\}$$

is positively invariant for (2). Therefore, we will study the global stability of (2) on the set Ω.

It is easily seen that the model (2) always has an infection-free equilibrium, $P_0 = (S_0, V_0, 0, 0, 0)$, where $S_0 = \frac{\Pi(\mu+\gamma-p\mu)}{\mu(\mu+\gamma)}$, $V_0 = \frac{p\Pi}{\mu+\gamma}$. Let $x = (E, I, V, S)^T$, then the Model (2) can be represented as

$$\frac{dx}{dt} = \mathcal{F}(x) - \mathcal{V}(x),$$

where

$$\mathcal{F}(x) = \begin{pmatrix} \frac{\beta SI}{1+\eta I} + \sigma\beta VI \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\mathcal{V}(x) = \begin{pmatrix} (\mu+\omega)E \\ (\mu+\alpha+\delta)I - \omega E \\ \sigma\beta VI + (\mu+\gamma)V - p\Pi \\ \frac{\beta SI}{1+\eta I} + \mu S - (1-p)\Pi - \gamma V \end{pmatrix}$$

Differentiating $\mathcal{F}(x)$ and $\mathcal{V}(x)$ with respect to E, I, V, S and computing them at the infection-free equilibrium $P_0 = (\frac{\Pi(\mu+\gamma-p\mu)}{\mu(\mu+\gamma)}, \frac{p\Pi}{\mu+\gamma}, 0, 0)$, respectively, we obtain

$$D\mathcal{F}(P_0) = \begin{pmatrix} 0 & \beta S_0 + \sigma\beta V_0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

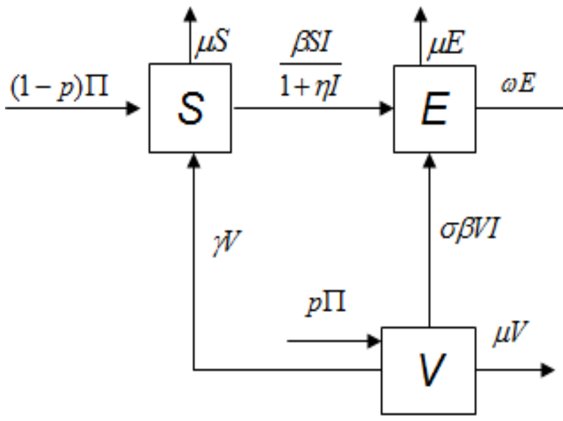


Figure 1: State transition diagram of the SVEIR model

$$D\mathcal{V}(P_0) = \begin{pmatrix} \mu+\omega & 0 & 0 & 0 \\ -\omega & \mu+\alpha+\delta & 0 & 0 \\ 0 & \sigma\beta V_0 & \mu+\gamma & 0 \\ 0 & -\beta S_0 & -\gamma & \mu \end{pmatrix}$$

Thus, the spectral radius of the next generation matrix $\mathcal{F}\mathcal{V}^{-1}$ can be written as,

$$\rho(\mathcal{F}\mathcal{V}^{-1}) = \frac{\omega\beta(S_0+\sigma V_0)}{(\mu+\omega)(\mu+\alpha+\delta)}.$$

According to Theorem 2 in [17], the basic reproduction number of the Model (2) is

$$\begin{aligned} R_0 &= \frac{\omega\beta(S_0+\sigma V_0)}{(\mu+\omega)(\mu+\alpha+\delta)} \\ &= \frac{\omega\beta\Pi(\frac{\mu+\gamma-p\mu}{\mu(\mu+\gamma)}+\frac{\sigma p}{\mu+\gamma})}{(\mu+\omega)(\mu+\alpha+\delta)}. \end{aligned} \quad (3)$$

3 Stability Analysis for Equilibriums

The endemic equilibrium $P^*(S^*, V^*, E^*, I^*)$ of the Model (2) can be obtained by the following Equations (4)

$$\begin{cases} (1-p)\Pi - \frac{\beta SI}{1+\eta I} - \mu S + \gamma V = 0, \\ p\Pi - \sigma\beta VI - (\mu+\gamma)V = 0, \\ \frac{\beta SI}{1+\eta I} + \sigma\beta VI - (\mu+\omega)E = 0, \\ \omega E - (\mu+\alpha+\delta)I = 0. \end{cases} \quad (4)$$

From the fourth equation of the Model (4), we can obtain $E = \frac{(\mu+\alpha+\delta)I}{\omega}$. Substituting E into the third equation of the Model (4), we can obtain $V = \frac{(\mu+\omega)(\mu+\alpha+\delta)}{\omega\sigma\beta} - \frac{\beta S}{(1+\eta I)\sigma\beta}$. According to the second equation of the Model (4), we obtain $I = \frac{p\Pi-(\mu+\gamma)V}{\sigma\beta V}$.

Substituting V and I into the first equation of Model (4) with the foregoing obtained values, we obtain the following equation

$$(1-p)\Pi - \frac{\beta S\frac{p\Pi-(\mu+\gamma)V}{\sigma\beta V}}{1+\eta I} - \mu S + \gamma A_1,$$

where, $A_1 = \frac{(\mu+\omega)(\mu+\alpha+\delta)}{\omega\sigma\beta} - \frac{\beta S}{(1+\eta I)\sigma\beta}$.

By a simple computation, we have

$$\Pi - \mu S - \frac{p\Pi(\mu+\omega)(\mu+\alpha+\delta)}{\frac{\omega\beta S}{1+\eta I} - (\mu+\omega)(\mu+\alpha+\delta)} + \frac{\mu S}{\sigma(1+\eta I)} + A_2 = 0,$$

where, $A_2 = \frac{\gamma(\mu+\omega)(\mu+\alpha+\delta)}{\omega\sigma\beta}$. Supposing

$$\begin{aligned} F(S) &= \Pi - \mu S - \frac{p\Pi(\mu+\omega)(\mu+\alpha+\delta)}{\frac{\omega\beta S}{1+\eta I} - (\mu+\omega)(\mu+\alpha+\delta)} \\ &\quad + \frac{\mu S}{\sigma(1+\eta I)} + \frac{\gamma(\mu+\omega)(\mu+\alpha+\delta)}{\omega\sigma\beta}. \end{aligned}$$

For $S = 0$, $F(0) = (1-p)\Pi + \frac{\gamma(\mu+\omega)(\mu+\alpha+\delta)}{\omega\sigma\beta}$. It is easily seen that $F(0) > 0$.

$$\begin{aligned} F^{'}(S) &= -\mu - \frac{p\Pi\omega\beta(\mu+\omega)(\mu+\alpha+\delta)\frac{1}{1+\eta I}}{(\frac{\omega\beta S}{1+\eta I} - (\mu+\omega)(\mu+\alpha+\delta))^2} + \frac{\mu}{\sigma(1+\eta I)} \\ &< -\mu + \frac{p\Pi\omega\beta\frac{1}{1+\eta I}}{\frac{\omega\beta S}{1+\eta I} - (\mu+\omega)(\mu+\alpha+\delta)} + \frac{\mu}{\sigma(1+\eta I)} \\ &= -\mu + (\frac{\mu}{\sigma} - \frac{p\beta\Pi}{\frac{(\mu+\omega)(\mu+\alpha+\delta)}{\omega} - \frac{\beta S}{1+\eta I}})\frac{1}{1+\eta I} \\ &< 0. \end{aligned}$$

Therefore, the sign of $F^{'}(S)$ is negative. On the other hand, if $R_0 > 1$, $\omega\beta(S_0+\sigma V_0) = \omega\beta(S_0 + \frac{\sigma p\Pi}{\mu+\gamma}) > (\mu +$

$\omega)(\mu+\alpha+\delta)$.

$$\begin{aligned}F(S_0) &= \Pi-\mu S_0-\frac{p\Pi(\mu+\omega)(\mu+\alpha+\delta)}{\frac{\omega\beta S_0}{1+\eta I}-(\mu+\omega)(\mu+\alpha+\delta)}\\ &\quad+\frac{\mu S_0}{\sigma(1+\eta I)}+\frac{\gamma(\mu+\omega)(\mu+\alpha+\delta)}{\omega\sigma\beta}\\ &< \Pi-\mu S_0-\frac{(\mu+\gamma)(\mu+\omega)(\mu+\alpha+\delta)}{\omega\beta\sigma}\\ &\quad+\frac{\mu S_0}{\sigma(1+\eta I)}+\frac{\gamma(\mu+\omega)(\mu+\alpha+\delta)}{\omega\sigma\beta}\\ &< \Pi-\mu S_0-\frac{\mu S_0}{\sigma(1+\eta I)}-\frac{\mu p\Pi}{\mu+\gamma}+\frac{\mu S_0}{\sigma(1+\eta I)}\\ &= \Pi-\mu S_0-\frac{\mu p\Pi}{\mu+\gamma}\\ &= 0.\end{aligned}$$

If $S > S_0$, $F(S) < 0$. As a result, the equation $F(S) = 0$ only has a root S^* which always exists in $(0, S_0)$. When $R_0 \le 1$, the System (2) only has an infection-free equilibrium $P_0(S_0, V_0, 0, 0)$. When $R_0 > 1$, the System (2) has the unique endemic equilibrium $P^*(S^*, V^*, E^*, I^*)$ except for the infection-free equilibrium P_0.

3.1 Infection-free Equilibrium and its Stability

It can be easily obtained that the Model (2) has an infection-free equilibrium given by $P_0 = (\frac{\Pi(\mu+\gamma-p\mu)}{\mu(\mu+\gamma)}, \frac{p\Pi}{\mu+\gamma}, 0, 0)$. The infection-free equilibrium corresponds to the model condition of non-worm breakout.

Proposition 1. *The infection-free equilibrium P_0 is locally asymptotically stable in the set Ω if $R_0 < 1$ and unstable if $R_0 > 1$.*

Proof. According to $P_0 = (\frac{\Pi(\mu+\gamma-p\mu)}{\mu(\mu+\gamma)}, \frac{p\Pi}{\mu+\gamma}, 0, 0)$, the Jacobian matrix at the infection-free equilibrium P_0 of the Model (2) is

$$J(P_0)=\begin{pmatrix}-\mu & \gamma & 0 & -\beta S_0\\ 0 & -\mu-\gamma & 0 & -\sigma\beta V_0\\ 0 & 0 & -\mu-\omega & \beta S_0+\sigma\beta V_0\\ 0 & 0 & \omega & -\mu-\alpha-\delta\end{pmatrix}$$

Therefore, the corresponding characteristic equation is described by

$$\begin{aligned}&(\lambda+\mu)(\lambda+\mu+\gamma)\\ &\quad\cdot[(\lambda+\mu+\omega)(\lambda+\mu+\alpha+\delta)-\omega(\beta S_0+\sigma\beta V_0)]\\ =&\ 0.\end{aligned}\tag{5}$$

From the characteristic Equation (5), we know that it always has two negative eigenvalues $\lambda_1 = -\mu$, and $\lambda_2 = -\mu-\gamma$. The other eigenvalues are decided by the following equation

$$(\lambda+\mu+\omega)(\lambda+\mu+\alpha+\delta)-\omega(\beta S_0+\sigma\beta V_0)=0. \tag{6}$$

By the simple computation, Equation (6) is equal to

$$\lambda^2+(2\mu+\omega+\alpha+\delta)\lambda+(\mu+\omega)(\mu+\alpha+\delta)-\omega(\beta S_0+\sigma\beta V_0)=0. \tag{7}$$

If $R_0 < 1$, $(\mu+\omega)(\mu+\alpha+\delta)-\omega(\beta S_0+\sigma\beta V_0) > 0$, thus two roots of Equation (7) are negative. The infection-free equilibrium P_0 to be locally asymptotically stable is that $\lambda_i < 0$, for $i = 1, 2, 3, 4$, which meets the sufficient condition of the stability theory [16]. When $R_0 > 1$, $(\mu+\omega)(\mu+\alpha+\delta)-\omega(\beta S_0+\sigma\beta V_0) < 0$, which means that $J(P_0)$ has a positive root and a negative root. Therefore, the infection-free equilibrium P_0 is an unstable saddle point. This completes the proof. □

Proposition 2. *When $R_0 \le 1$, the infection-free equilibrium P_0 is globally asymptotically stable in the set Ω.*

Proof. To prove the infection-free equilibrium P_0 is globally asymptotically stable, we construct the following Lyapunov function: $L(E, I) = \omega E + (\mu+\omega)I$.

Its derivative along the solutions to the Model (2) is

$$\begin{aligned}L'(t) &= \omega E'+(\mu+\omega)I'\\ &= \frac{\omega\beta SI}{1+\eta I}+\omega\sigma\beta VI-\omega(\mu+\omega)E+(\mu+\omega)\omega E\\ &\quad-(\mu+\omega)(\mu+\alpha+\delta)I\\ &= \frac{\omega\beta SI}{1+\eta I}+\omega\sigma\beta VI-(\mu+\omega)(\mu+\alpha+\delta)I\\ &\le (\omega\beta S+\omega\sigma\beta V-(\mu+\omega)(\mu+\alpha+\delta))I\\ &= \frac{\omega\beta(S_0+\sigma V_0)}{R_0}(\frac{R_0(S+\sigma V)}{S_0+\sigma V_0}-1)I\\ &\le 0.\end{aligned}$$

Furthermore, $L' = 0$ if and only if $I = 0$. Thus, the largest compact invariant set in $\{(S, V, E, I)|L' = 0\}$ is the singleton P_0. When $R_0 \le 1$, the global stability of P_0 satisfies LaSalle's invariance principle [8]. LaSalle's invariance principle [8] hints that P_0 is globally asymptotically stable in the set Ω. This completes the proof. □

3.2 Endemic Equilibrium and Its Stability

From the aforementioned computation, we know that the Model (2) has the unique endemic equilibrium P^*. The endemic equilibrium P^* means that the worm does not die out when it appears. Finally, every class of the model reach its stable state. S^*, V^*, E^*, I^* and R^* are not equal to zero. Next, we investigate the local stability of the endemic equilibrium $P^* = (S^*, V^*, E^*, I^*)$.

Proposition 3. *When $R_0 > 1$, the endemic equilibrium P^* is locally asymptotically stable in the region Ω.*

Proof. The Jacobian matrix of (2) at the endemic equilibrium P^* is

$$J(P^*) = \begin{pmatrix} -B_1 & \gamma & 0 & -\frac{\beta S^*}{(1+\eta I^*)^2} \\ 0 & -B_2 & 0 & -\sigma\beta V^* \\ \frac{\beta I^*}{1+\eta I^*} & \sigma\beta I^* & -(\mu+\omega) & B_3 \\ 0 & 0 & \omega & -(\mu+\alpha+\delta) \end{pmatrix}$$

where,

$$\begin{aligned} B_1 &= \frac{\beta I^*}{1+\eta I^*} + \mu, \\ B_2 &= \sigma\beta I^* + (\mu+\gamma), \\ B_3 &= \frac{\beta S^*}{(1+\eta I^*)^2} + \sigma\beta V^*. \end{aligned}$$

Thus, the corresponding characteristic equation can be described as

$$\lambda^4 + C_1\lambda^3 + C_2\lambda^2 + C_3\lambda + C_4 = 0, \quad (8)$$

where,

$$\begin{aligned} C_1 &= 4\mu + \alpha + \omega + \delta + \gamma + \sigma\beta I^* + \frac{\beta I^*}{1+\eta I^*} \\ &> 0, \\ C_2 &= (\mu+\omega)(\mu+\alpha+\delta) + B_2(2\mu+\omega+\alpha+\delta) \\ &\quad +B_1(\sigma\beta I^* + 3\mu + \gamma + \omega + \alpha + \delta) \\ &> 0, \\ C_3 &= B_2(\mu+\omega)(\mu+\alpha+\delta) + \beta\omega\mu\frac{S^*}{(1+\eta I^*)^2} \\ &\quad +B_1[(\mu+\omega)(\mu+\alpha+\delta) + B_2(2\mu+\omega+\alpha+\delta)] \\ &\geq (\mu+\omega)(\mu+\alpha+\delta)(\sigma\beta I^* + 2\mu + \gamma + \frac{\beta I^*}{1+\eta I^*}) \\ &\quad +B_1B_2(2\mu+\omega+\alpha+\delta) \\ &> 0, \\ C_4 &= B_1B_2(\mu+\alpha+\delta) + \gamma\omega\mu\sigma\beta V^* \\ &\quad +\beta\omega\mu\frac{S^*}{(1+\eta I^*)^2}B_2 \\ &\geq B_1B_2(\mu+\omega)(\mu+\alpha+\delta) + \gamma\omega\mu\sigma\beta V^* \\ &> 0. \end{aligned}$$

Through a simple computation, we obtain that $H_1 = C_1 > 0$, $H_2 = C_1C_2 - C_3 > 0$,$H_3 = C_3H_2 - C_1^2C_4 > 0$, $H_4 = C_4H_3 > 0$.

According to the theorem of Routh-Hurwitz [10], we obtain that all the roots of the Equation (8) have negative real parts. As a result, the endemic equilibrium P^* is locally asymptotically stable. □

4 Numerical Simulations

This section develops numerical experimental steps to analyze the stability of the proposed model and evaluates the effects of the implemented countermeasures. It is very difficult to use realistic parameters or real-world traffic traces for our research, because many parameters used in previous models are assumed according to their hypothesis. To obtain the spread of worms in a large-scale network, $N = 1,000,000$ hosts are selected as the population size. According to the real Slammer worm, the average scan rate is $s = 4,000$ per second [11]. The infection rate of the Slammer worm is $\beta = s/2^{32} = 9.3 \times 10^{-7}$. We take proper values of Π and μ so that $\Pi/\mu = N$, implying that the total number of hosts remain unchanged. Therefore, we set $\Pi = 100$ and $\mu = 0.0001$. The partial immunization rate is set to $\sigma = 0.4$. The transition rate ω from E to I is 0.02. The transition rate δ from I to R is 0.01. At the beginning, the number of susceptible, vaccinated, exposed, infected, and recovered hosts are $S(0) = 999,985$, $V(0) = 10$, $E(0) = 0$, $I(0) = 5$, and $R(0) = 0$, respectively.

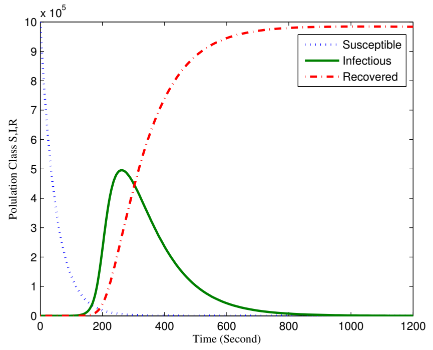


Figure 2: Globally asymptotically stable infection-free equilibrium

Other parameters in these simulations are given as follows: $\eta = 2$, $p = 0.2$, $\alpha = 0.0001$, and $\gamma = 0.05$. The results are based on the average of at least 10 simulation runs. Using the above parameters, we can obtain the basic reproduction number $R_0 = 0.908 < 1$. The worm will gradually die out according to Proposition 1 and 2. Figure 2 illustrates the number of susceptible, infected and recovered hosts when R_0 is 0.908, respectively. From Figure 2, we can clearly see that the tendency of the worm propagation is depressive, which is consistent with Proposition 1 and 2. Finally, all infected hosts vanish and the population, in the long term, is in a recovered state.

In the second experiment, we change some related parameters about R_0 to guarantee $R_0 > 1$. When $p = 0.4$ and $\delta = 0.003$, we have $R_0 = 9.847 > 1$. Other parameters remain unchanged. The results are shown in Figure 3. As can be seen from Figure 3, the number of susceptible, infected and recovered hosts eventually become positive values between 0 and Π/μ, which indicates that the worm does not disappear, if worms initially present. Finally, these three states reach their equilibrium points $P^*(38094, 83543, 835428)$. This is fully consistent with the conclusions of Proposition 3. The unique endemic equilibrium P^* is globally asymptotically stable.

To demonstrate that the effect of the partial immunization rate on the number of infected hosts, we set

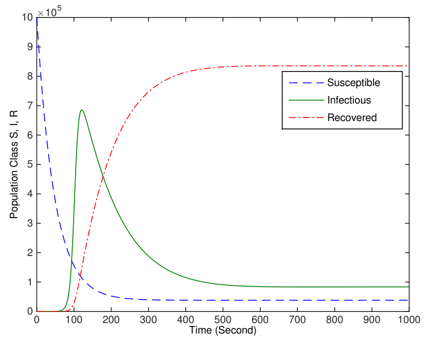


Figure 3: Locally asymptotically stable endemic equilibrium

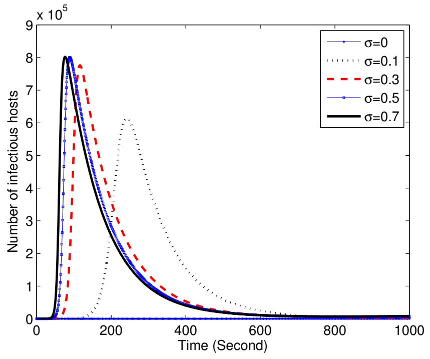


Figure 4: Effect of the partial immunization rate

the partial immunization rate σ to different values, with other parameters remaining the same. Figure 4 shows the effect of changing the partial immunization rate ($\sigma = 0, 0.1, 0.3, 0.5, 0.7$, respectively) on worm propagations. From Figure 4, no hosts are infected when $\sigma = 0$. $\sigma = 0$ means that all hosts gain full immunization. However, in real-world networks, it is very difficult to implement full immunization. As expected, a smaller partial immunization rate results in slowing down the worm propagation speed, more importantly, and decreasing the total number of infected hosts. Once the vaccine has been studied, computer users should immunize their computers as quickly as possible, which can guarantee to reach a smaller partial immunization rate σ.

5 Conclusion

This paper presented a novel dynamic *SVEIR* model with a saturated incidence rate and a partial immunization rate for the propagation of worms. More specifically, this paper investigated the global dynamic behavior of the *SVEIR* model, which is determined by the basic reproduction number. The theoretical analysis demonstrated that when the basic reproduction number is smaller than or equal to one, the *SVEIR* model has a infection-free equilibrium, and is globally asymptotically stable. That is to say, it implies that the worm dies out eventually. When the basic reproduction number is larger than one, the *SVEIR* model has a unique endemic equilibrium which is locally stable. Moreover, it implies that worms are able to pervade across networks. The simulation results are consistent with theoretical analysis. Our proposed *SVEIR* model will be highly useful to analyze the availability and efficiency of partial immunization. The partial immunization will be efficient if the partial immunization rate is very small.

Acknowledgments

The authors gratefully acknowledge the erudite comments and suggestions of the anonymous reviewers, which have improved the presentation. This research was supported by the National Natural Science Foundation of China under (No. 61272541, 61572170), Natural Science Foundation of HeBei Province of China under (No. F2015205157), Natural Science Foundation of Hebei Normal University of China under (No. L2015Z08), Educational Commission of Hebei Province of China under (No. QN2014165).

References

[1] J. L. Aron, I. B. Schwartz, “Seasonality and period-doubling bifurcations in an epidemic model,” *Journal of Theoretical Biology*, Vol. 110, no. 4, pp. 665–679, 1984.

[2] L. Cai, A. A. Lashari, I. H. Jung, K. O. Okosun, Y. I. Seo, “Mathematical analysis of a malaria model with partial immunity to reinfection,” *Abstract and Applied Analysis*, vol. 2013, Article ID 405258, pp. 1–17, 2013.

[3] V. Capasso, G. Serio, “A generalization of the kermack-mckendrick deterministic epidemic model,” *Mathematical Biosciences*, vol. 42, no. 1, pp. 43–61, 1978.

[4] C. Feng, J. Yang, Z. Qin, D. Yuan, H. Cheng, “Modeling and analysis of passive worm propagation in the P2P file-sharing network,” *Simulation Modelling Practice and Theory*, vol. 51, no. 2, pp. 87–99, 2015.

[5] C. Gan, X. Yang, W. Liu, Q. Zhu, X. Zhang, “An epidemic model of computer viruses with vaccination and generalized nonlinear incidence rate,” *Applied Mathematics and Computation*, vol. 222, no. 3, pp. 265–274, 2013.

[6] C. Gan, X. Yang, W. Liu, Q. Zhu, “A propagation model of computer virus with nonlinear vaccination probability,” *Communications in Nonlinear Science and Numerical Simulation*, vol. 19, no. 1, pp. 92–100, 2014.

[7] W. Kermack, A. McKendrick, “Contributions of mathematical theory to epidemics,” *Proceedings of the Royal Society of London: Series A*, Vol. 115, no. 772, pp. 700–721, 1927.

[8] J. P. LaSalle, *The stability of dynamical systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1976.

[9] B. K. Mishra, S. K. Pandey, "Dynamic model of worm propagation in computer network," *Applied Mathematical Modelling*, vol. 38, no. 7, pp. 2173–2179, 2014.

[10] J. Morris, "The routh and routh-hurwitz stability criteria," *Aircraft Engineering and Aerospace Technology*, vol. 34, no. 1, pp. 25–27, 1962.

[11] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, N. Weaver, "Inside the slammer worm," *IEEE Security and Privacy*, vol. 1, no. 4, pp. 33–39, 2003.

[12] M. Ozair, T. Hussain, "Analysis of vector-host model with latent stage having partial immunity," *Applied Mathematical Sciences*, vol. 8, no. 32, pp. 1569–1584, 2014.

[13] S. Peng, M. Wu, G. Wang, S. Yu, "Propagation model of smartphone worms based on semi-markov process and social relationship graph," *Computers & Security*, vol. 44, no. 1, pp. 92–103, 2014.

[14] M. Peltomäki, M. Ovaska, M. Alava, "Worm spreading with immunization: An interplay of spreading and immunity time scales," *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 23, pp. 4152–4159, 2011.

[15] J. Ren, Y. Xu, Y. Zhang, Y. Dong, G. Hao, "Dynamics of a delay-varying computer virus propagation model," *Discrete Dynamics in Nature and Society*, vol. 2012, Article ID 371792, pp. 1–12, 2012.

[16] R. C. Robinson, *An introduction to dynamical systems: continuous and discrete*, American Mathematical Society, Charles Street, Providence, RI, USA, 2012.

[17] P. Van den Driessche, J. Watmough, "Reproduction numbers and subthreshold endemic equilibria for compartmental models of disease transmission," *Mathematical Biosciences*, vol. 180, no. 1, pp. 29–48, 2002.

[18] S. Wang, Q. Liu, X. Yu, Y. Ma, "Bifurcation analysis of a model for network worm propagation with time delay," *Mathematical and Computer Modelling*, vol. 52, no. 3, pp. 435–447, 2010.

[19] Y. Yao, X. Xie, H. Guo, G. Yu, F. Gao, X. Tong, "Hopf bifurcation in an 320 internet worm propagation model with time delay in quarantine," *Mathematical and Computer Modelling*, vol. 57, no. 11–12, pp. 2635–2646, 2013.

[20] Y. Yao, X. Feng, W. Yang, W. Xiang, F. Gao, "Analysis of a delayed internet worm propagation model with impulsive quarantine strategy," *Mathematical Problems in Engineering*, vol. 2014, Article ID 369360, pp. 1–18, 2014.

[21] L. Yang, X. Yang, "The effect of infected external computers on the spread of viruses: A compartment modeling study," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 24, pp. 6523–6535, 2013.

Fangwei Wang received his B.S. degree in 2000 from College of Mathematics & Information Sciences, Hebei Normal University, his M.S. degree in 2003 from College of Computer Science and Software, Hebei University of Technologyis, his Ph.D degree in 2009 from College of Computer at Xidian University. Currently he is an associate professor at Hebei Normal University, Shijiazhuang, China. His research interests include: network and information security, sensor networks.

Hongfeng Gao received his B.S. degree in 1996 from College of physics and Electronic Science, Guizhou University, his M.S. degree in 2007 from College of Computer Science, Guizhou University. Currently he is an associate professor at Guizhou University, Guiyang, China. His research interests include network and information security.

Yong Yang received his B.S. degree in 1998 from Department of Information and Electronic Science, Yunnan University, his M.S. degree in 2003 from School of Information Science and Engineering, Yunnan University. Currently he is a lecturer at Yunnan University, Kunming, China. His research interests include network and information security.

Changguang Wang received his M.S. degree in 1996 from School of Physical Science and Technology, Sichuan University, and his Ph.D degree in 2009 from College of Computer at Xidian University. Currently he is a professor at Hebei Normal University, Shijiazhuang, China. His research interests include network and information security.

Integrating the Functional Encryption and Proxy Re-cryptography to Secure DRM Scheme

Hisham Abdalla, Xiong Hu, Abubaker Wahaballa, Ahmed Abdalla,
Mohammed Ramadan and Qin Zhiguang
(Corresponding author: Hisahm Abdalla Sedahmed)

University of Electronic Science and Technology of China
2006 Xiyuan Avenue, Gaoxin West Zone, Chengdu 611731, China.
(Email: hisham_awaw@hotmail.com)
(Received Nov. 03, 2015; revised and accepted Jan. 23, 2016)

Abstract

The current Digital Rights Management (DRM) systems use attribute-based encryption (ABE) and proxy re-encryption (PRE) to achieve fine-grained access control in cloud computing. However, these schemes have some limitations particularly in terms of security, functionality and also higher decryption time which grows linearly with the complexity of access policies. In this paper, we propose a novel DRM scheme founded on a deterministic finite automata-based functional proxy re-encryption (DFA-based FPRE) scheme which has been proven to be secure against CCA in the standard model. In particular, we leverage the DFA-based FPRE scheme to realize fine-grained access control over encrypted contents among a set of users. Furthermore, a secure content key distribution protocol and efficient revocation mechanism are provided. Moreover, we tackle the critical issue of high computation at the user side, by outsourcing computation into (DFA-based FPRE) scheme for the first time. In comparison, our scheme achieves higher efficiency and smaller computation time against state-of-the-art.

Keywords: Cloud computing, digital rights management, fine-grained access control, privacy preserving

1 Introduction

The rapid development and growth of the Internet have fuelled a trend towards outsourcing data and its management. The excitement of the emerging technology is due to the advancement of internet, whose infrastructure is cloud computing. It brings a flexible, cost effective and reliable way for data owners to deal with their data storage. Storing digital contents to the cloud, enable users to concentrate on their core business issues rather than incurring substantial hardware, software, or personal costs.

However, owners still have to remain cautious to protect their contents from being pirated and illegally distributed [24]. The cloud service provider is semi-trusted [27]. In this sense, the semi-trusted cloud service provider follows the normal flow of the protocol in the system. For instance, during the interaction with the users, a CSP may collect users' personal information and consumption profiles, which inspires a serious security concern for cloud user. Proxy re-encryption (PRE) technique [23] is devised to prevent the CSP from accessing the contents in semi-trusted cloud environment. Also, it's crucial for the CSP to be prevented from knowing exactly which users are accessing certain contents [21].

Digital rights management (DRM) is a famous mechanism to protect content copyright [1] based on the techniques of content encryption, access control, and dynamic licensing [16, 31]. In the past few years, there have been some DRM schemes which deal with confidentiality and privacy preserving of outsourced data in cloud computing. Petrlic et al. [23] introduced a privacy-preserving cloud DRM scheme based on proxy re-encryption, which allows a user anonymously purchase content from a content provider, and in the same time prevents any party from building usage profiles under a pseudonym. Petrlic also presented a privacy-preserving DRM scheme [21], which employs a combination of ring signatures with an anonymous recipient scheme. Secret sharing makes it possible for the content provider to expose the user identities in case of fraud. Perlman et al. proposed a privacy-preserving DRM conception on the basis of anonymous cash and blind decryption. Their scheme allows users to buy digital content without exposing their track [22]. Although these schemes are able to ensure data security, these schemes cannot support fine-grained access control, or limit a set of individual users to access encrypted data.

In order to solve these problems, Muller et al. proposed a new DRM architecture which limits the digital content access to a subset of users who possess certain properties assigned during the encryption process [17]. In their model, the set of rules are divided into two part, static and dynamic. The static rules are enforced by using ABE before accessing the content, while dynamic rules stored in the license needs to be enforced at run time by the DRM viewer. However, revocation cannot be achieved in this scheme. What's worse is that, this scheme is not applicable for large numbers of users, which may be a huge burden for server.

As a result, attempting to achieve the revocation mechanism, the traditional revocation schemes relying on attribute authority would usually enforce periodically re-encrypt content, and re-generate a new secret keys to legal users as in [8, 30]. However, these schemes always results in key update operation. In practice, a large number of users can access cloud services. Hence, these schemes are far from have been suitable in cloud. Being aware of this problem, this paper follow schemes [4, 5], which permits the delegated key server to revoke the attributes and the malicious users immediately.

Thus, the ABE and PRE are usually employed to solve the problems above. Nevertheless, security and functional problems still exist. Additionally, besides [4, 5, 17], these schemes still suffer from the drawback of high computational cost associated with ABE operations; that is, both the computational cost and the ciphertext size for the users grow linearly with the size of the access formula. Hence, in this paper, we adopt a deterministic finite automata-based functional proxy re-encryption (DFA-based FPRE) scheme [13] to protect the contents stored in the semi-trusted cloud environment. Besides, outsourcing computation into (DFA-based FPRE)to avoid the high computation at the user side.

1.1 Motivation

Although using ABE and PRE can solve practical network problems, this method leaves interesting open problems in terms of security and functionality. As to security, it is not easy for ABE constructions [9, 10, 19, 20] to achieve adaptive security without random oracles [25]. Meanwhile, all existing attribute-based PRE (ABPRE) schemes [11, 12, 15] are proven secure only against chosen-plaintext attacks (CPA) in the selective model, whereas security against chosen ciphertext attack (CCA) is considered an important notion for ABPRE schemes. The functionality of an ABPRE system is another practical issue. All existing ABPRE schemes only support access policy combining with AND gates and fixed size number of boolean variables inputs. Practically, an access policy might be required to combine with AND, OR gates and NOT. Also, in some particular applications, the access policy might be expressed by regular languages with arbitrary size input data.

Referring to the research above, our DRM scheme is founded on a DFA-based FPRE scheme [13]. Having proved to be secure against CCA in the standard model, the DFA-based FPRE scheme also provides unlimited size input for access policy while the functionality of proxy re-encryption remains still.

In a nutshell, our contribution can be summarized as:

1) We propose a secure key management mechanism in DRM based on DFA-based FPRE scheme.

2) We introduce a fine-grained access control mechanism, which allows flexibility in specifying the access rights of individual users.

3) Our scheme provides a scalable revocation mechanism, which allows the delegated key server in the cloud to revoke the attributes and malicious users immediately.

4) We perform an analysis evaluation of our DFA-based FPRE DRM scheme.

The rest of this paper is organized as follows. In next Section the preliminaries required in this paper are presented. Our DRM scheme based on DFA-based FPRE scheme is presented in Section 3. Analysis of our scheme is discussed in Section 4. Finally, the conclusion is introduced in Section 5.

2 Preliminaries

Our scheme relies on a DFA-based functional proxy re-encryption scheme. We will briefly introduce the DFA-based functional proxy re-encryption scheme and the groups underlying our encryption scheme.

2.1 Composite Order Bilinear Groups

Let $\mathbb{G}$ and $\mathbb{G}_T$ be multiplicative cyclic groups of same order $N = P_1P_2P_3$ (where P_1, P_2, P_3 are distinct primes). We call a map $e : \mathbb{G}\times\mathbb{G} \longrightarrow \mathbb{G}_T$ bilinear if it should satisfy the following properties:

- Bilinear. $e(g^a, h^b) = e(g, h)^{ab}$, $\forall g, h \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z_N}^*$; and
- Non-degenerate. There exists $g \in \mathbb{G}$ such that $e(g, g)$ is a generator of $\mathbb{G}_T$.

We denote by G_{p_1}, G_{p_2} and G_{p_3} the subgroups of $\mathbb{G}$ of respective orders p_1, p_2 and p_3.

2.2 Complexity Assumptions

Definition 1. *(*The Source Group l-Expanded Bilinear Diffie-Hellman Exponent (l-Expanded BDHE.) Assumption in a Subgroup [13]*). Given a group generator $\mathscr{G}$ and a positive integer l, we define the following distribution:*

$$\begin{aligned}(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) &\longleftarrow \mathscr{G},\\ g_1 &\overset{R}{\longleftarrow} \mathbb{G}_{p_1},\\ g_2 &\overset{R}{\longleftarrow} \mathbb{G}_{p_2},\\ g_3 &\overset{R}{\longleftarrow} \mathbb{G}_{p_3},\\ a, b, d, m, n, x, c_0, \cdots, c_{l+1} &\overset{R}{\longleftarrow} \mathbb{Z}_N,\end{aligned}$$

Table 1: Notations in proposed scheme

Notion	Description	Notion	Description
K	security parameter	PP, MSK	public parameters and master key
U	user	CP	content provider
SP	service provider	LS	license server
CSP	cloud service provider	PK_U, SK_U	public and secret keys of user
PK_{CP}	public key of content provider	$RK_{M\to w}$	re-encryption key
CMK	content master key	AK	assistant key
CEK	content encryption key	CID	content identity
M	plain content data	CT	encrypted content data
UR	user rights	RE	rights expression
T	timestamp	$\sigma()$	signature algorithm
σ_{LS}	license acquisition request signature	σ_{KS}	key acquisition request signature
σ_L	license signature	SK_{CP}	secret key of content provider

$$\begin{aligned}
D &= (N, \mathbb{G}, \mathbb{G}_T, e, g_1, g_2, g_3, g_2^a, g_2^b, g_2^{ab/dx}, g_2^{b/dx}, g_2^{ab/x},\\
&\quad g_2^n, \forall i \in [0, 2l+1], i \neq l+1, j \in [0, l+1] g_2^{a^i mn},\\
&\quad g_2^{a^i bmn/c_j x}, \forall i \in [0, l+1] g_2^{c_i}, g_2^{a^i d}, g_2^{abc_i/dx}, g_2^{bc_i/dx},\\
&\quad \forall i \in [0, 2l+1], i \neq l+1, j \in [0, l+1] g_2^{a^i bd/c_j x},\\
&\quad \forall i, j \in [0, l+1], i \neq j g_2^{a^i bc_j/c_i x}),\\
T_0 &= g_2^{a^{l+1} bm},\\
T_1 &\xleftarrow{R} \mathbb{G}_{p_2}.
\end{aligned}$$

The advantage of an algorithm $\mathscr{A}$ in breaking this assumption is $Adv_{\mathscr{A}}^{l-BDHE}(1^n) =\mid Pr[\mathscr{A}(D, T_0) = 1] - Pr[\mathscr{A}(D, T_1) = 1] \mid$. We say that $\mathscr{G}$ satisfies the l-Expanded BDHE Assumption if $Adv_{\mathscr{A}}^{l-BDHE}(1^n)$ is negligible for any PPT algorithm $\mathscr{A}$.

Definition 2. *(*The Source Group Modified q Bilinear Diffie-Hellman Exponent (q-BDHE) Assumption in a Subgroup [13].*) Given a group generator $\mathscr{G}$, we define the following distribution:*

$$\begin{aligned}
(N = p_1 p_2 p_3, \mathbb{G}, \mathbb{G}_T, e) &\longleftarrow \mathscr{G},\\
g &\xleftarrow{R} \mathbb{G}_{p_1},\\
g_2 &\xleftarrow{R} \mathbb{G}_{p_2},\\
g_3 &\xleftarrow{R} \mathbb{G}_{p_3},\\
c, a, e, f &\xleftarrow{R} \mathbb{Z}_N,
\end{aligned}$$

$$\begin{aligned}
D &= (N, \mathbb{G}, \mathbb{G}_T, e, g, g_2, g_3, g_2^e, g_2^a, g_2^{eaf}, g_2^{c+f/c}, g_2^{c^2}, \cdots,\\
&\qquad g_2^{c^q}, g_2^{1/ac^q}),\\
T_0 &= g_2^{aec^{q+1}},\\
T_1 &\xleftarrow{R} \mathbb{G}_{p_2}.
\end{aligned}$$

The advantage of an algorithm $\mathscr{A}$ in breaking this assumption is $Adv_{\mathscr{A}}^{q-BDHE}(1^n) =\mid Pr[\mathscr{A}(D, T_0) = 1] - Pr[\mathscr{A}(D, T_1) = 1] \mid$. We say that $\mathscr{G}$ satisfies the Source Group Modified q-BDHE Assumption if $Adv_{\mathscr{A}}^{q-BDHE}(1^n)$ is negligible for any PPT algorithm $\mathscr{A}$.

2.3 A DFA-based Functional Proxy Re-encryption Scheme

For more details we refer the reader to [28] for the definition of DFA and DFA-based FE. The DFA-based functional proxy re-encryption scheme consists of the following seven algorithms [13]:

1) $(PP, MSK) \leftarrow Setup(1^n, \Sigma)$: The system setup algorithm takes a security parameter n and the description of a finite alphabet Σ as input. It outputs the public parameters PP and a master key MSK, where $n \in \mathbb{N}$. Here, we note that PP implicitly includes Σ.

2) $SK_M \leftarrow k.Gen(MSK, M = (Q, \tau, q_0, F))$: The key generation algorithm takes the master key MSK and a DFA description M as input. It outputs a private key SK_M, where Q is a set of states, τ is a set of transitions, $q_0 \in Q$ is a start state and $F \subseteq Q$ is a set of accept states.

3) $RK_{M\to w} \leftarrow ReKeyGen(SK_M, w)$: This algorithm takes SK_M for a DFA description M and an arbitrary length string $w \in \sigma$ as input. It outputs a re-encryption key $RK_{M\to w}$. Using the re-encryption key any ciphertext under a string $w^{'}$ (in which $ACCEPT(M, w^{'})$), it can converted to another ciphertext under w.

4) $CT \leftarrow DFA.E(PP, w, m)$: The encryption algorithm takes the public parameters PP, a message m and a $w \in \Sigma$ as input. It outputs the ciphertext CT under w.

5) $CT^R \leftarrow ReEnc(Rk_{M\to w}, CT)$: The encryption algorithm takes $Rk_{M\to w}$ and CT (under $w^{'}$). If $ACCEPT(M, w^{'})$, it outputs the ciphertext CT^R under w.

6) $m/\perp \leftarrow DFA.D(SK_M, CT)$: The decryption algorithm takes a secret key SK_M and ciphertext CT

(under w) as input. The decryption can be done if $ACCEPT(M, w)$, then it outputs a message m; otherwise, outputs an error symbol $\perp$.

7) $m/\perp \leftarrow DFA.D_R(SK_M, CT^R)$: The decryption algorithm takes a secret key SK_M and ciphertext CT^R (under w) as input. The decryption can be done if $ACCEPT(M, w)$, then it outputs a message m; otherwise, outputs an error symbol $\perp$.

3 Proposed Scheme

3.1 Security Requirements

We provide the requirements of our proposed scheme into two terms namely security and privacy, as follows:

1) **Efficient**: In cloud computing the user is expected to access various contents through multiple devices any time anywhere without limitation, and also looks for flexible usage model. Therefore, the DRM scheme in cloud computing should provide efficient license distribution models with low computational complexity to support huge number of users.

2) **Security**: The content provider is supposed to ensure that an authorized user is not able to extract and run the content. Also, content confidentiality against unauthorized users must be achieved. Meanwhile, server provider and license server must not be able to get the plain content and content key.

3) **Privacy preserving**: To realize the user privacy preserving, the user should stay anonymous towards the content provider that deals with user's content purchase and the license server that receives acquisition request. Therefore, neither content provider nor license server will be able to retrieve user's personal information, such as user identity, IP address, etc.

4) **Collusion-resistance**: The group of non revoked yet unauthorized users should not be able to pull together their information (DFA) to decrypt an encrypted content in that each of them is unable to decrypt it individually.

3.2 Basic DRM System Model

The basic architecture of DRM consists of seven entities as shown in Figure 1 and the notations are shown in Table 1.

1) **Cloud storage**: This entity provides a storage service based on cloud computing, which holds the encrypted contents from the content providers.

2) **Key server**: It is an entity that generates the public/private key pair for content provider and user, and keeps the encrypted content master key and assistant key issued by content provider. Further, key server re-encrypts the assistant key to the license server when users acquire to consume the content.

3) **Public authority**: This entity generates the public parameters PP and a master key MSK for the system. It also works as a key authority and issues secret keys associated with DFA to users. A key server is delegated by the public authority to perform a revocation task, revoking DFA of user and illegal users immediately. In addition, it allows flexibility in specifying the access rights of individual users according to their attributes description.

4) **Cloud service provider**: This entity keeps the encrypted content in the cloud storage. It is in charge of computing the transformed data, providing corresponding encrypted contents and license distribution to the user.

5) **License server**: This entity generates and distributes the license for authorized users whenever receiving the license acquisition from the CSP. The license includes the encrypted CMK.

6) **Content provider**: This is an entity that holds the digital contents and protect the contents from unauthorized user by encrypting their own contents with the content encryption key. Then, content providers outsource their encrypted contents to cloud storage provided by the CSP.

7) **User**: This is an entity that can get the encrypted content from the CSP. If a user owns the DFA that is satisfying the string w of the ciphertext, he will be able to recover the content encryption key. Then, he can decrypt and play the contents.

3.3 Intuition

In order to achieve a secure DRM scheme, we leverage the DFA-based FPRE scheme as the basic cryptographic tool and combine the outsourcing techniques and efficient revocation to tackle the focusing issues on efficiency, immediate revocation and fine-gained access control.

Our proposed construction operates as follows:

1) In system setup and key generation, the public authority generates the public parameters PP and a master key MSK. It also generates a user DFA and secret keys SK_M denoted as $SK_M = (M, \epsilon, \Omega, DK)$ for each user. Further, it also generates the re-encryption key $Rk_{M\rightarrow w}$ for authorized user and sends it to the key server in secure channel. Moreover, the content provider generates the CEK with random CMK and AK.

2) In content packaging and encryption, the content provider encrypts the contents C_x with content encryption key CEK_x, It then gets the encrypted contents in the following form:

$$E_{sym}(C_x|CEK_x), \quad \text{where} \quad x = 1, 2, 3, .., n$$

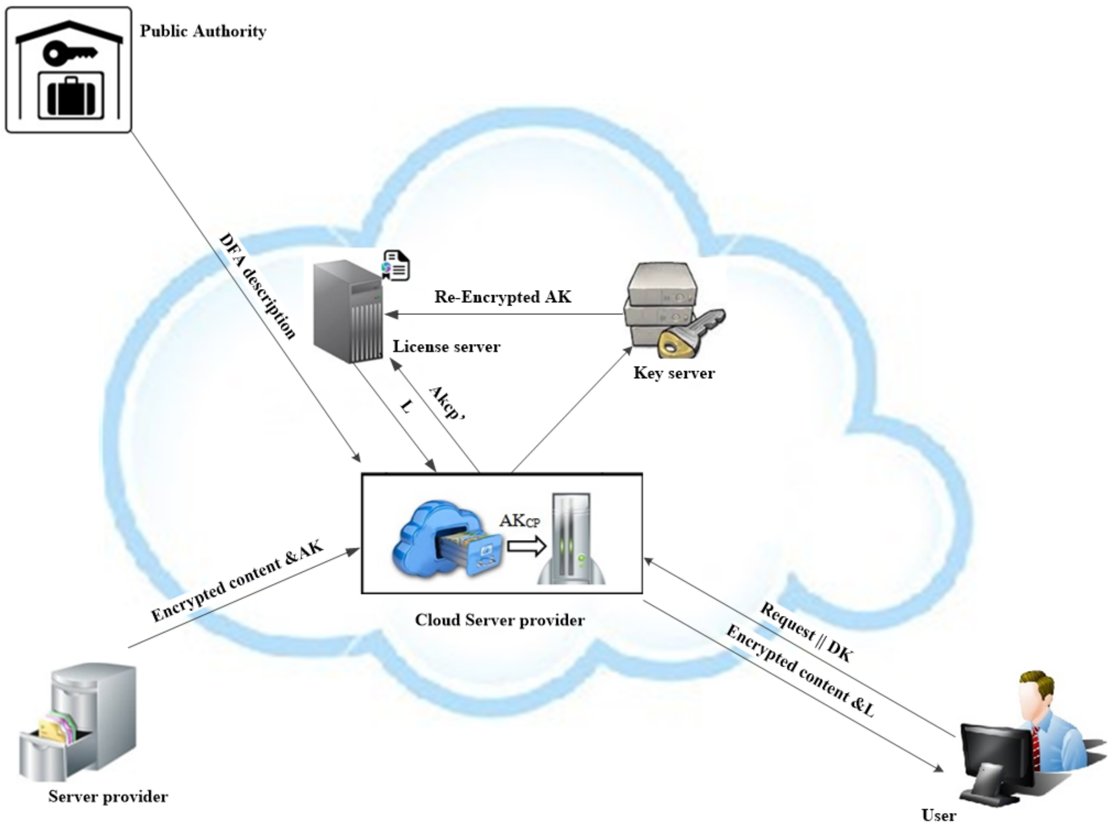


Figure 1: The system model of the proposed scheme system

Content provider later outsources the encrypted content to content server provider. Padding is employed to the contents before the encryption to make sure that each content has the same length. The sequence diagram of content encryption steps are shown in Figure 2.

3) In license acquisition, the user chooses the interesting content C_x with a unique identifier CID from the CSP, which is allowed to download the encrypted content. After downloading the DFC header from the content service provider, the user extracts the CMK_M from the DFC header and make sure that his/her DFA satisfy the string w of the content. A user cannot play the content without the valid license. Meanwhile, in order to acquire the license, the user first sends his partial decryption key DK to the content service provider, which is part of his private key. In fact, he just needs to send it once, unless his private key is regenerated, then content service provider transforms the ciphertext AK_{CP} to $AK_{CP}{'}$ and finally sends license acquisition request ($LSQ = \{CID \parallel UR \parallel T \parallel \sigma_{LS} \parallel AK_{CP}{'}\}$) including the user's rights UR, CID, T, $AK_{CP}{'}$ and $\sigma_{LS} = \sigma(SK_{CSP}, CID \parallel UR \parallel T \parallel AK_{CP}{'})$ to license server.

Upon receiving the user's license acquisition request, the license server checks the signature σ_{LS} and T, and then acquires the assistant key from the key server. The key acquisition request includes $\{CID \parallel T \parallel \sigma_{KS}\}$, where $\sigma_{KS} = \sigma(SK_{LS}, CID \parallel UR \parallel T)$. The key server checks the signature σ_{KS} and T. Then, key server computes the re-encrypted (AK^R_{CP}) and sends it to the license server. After that, the license server generates the right expression RE from the UR according to the right expression language and also generates the license $L = \{CID \parallel RE \parallel AK^R_{CP} \parallel \sigma_L \parallel AK_{CP}{'}\}$, which includes content identity CID, AK^R_{CP}, $AK_{CP}{'}$, right expression UE and signature $\sigma_L = \sigma(SK_{LS}, CID \parallel AK^R_{CP} \parallel AK_{CP}{'} \parallel RE)$. Finally, the license server sends L to user through CSP. Upon receiving L, user checks the signature and keeps the license.

4) In content consumption, whenever a user want to play the content, the user will compute the content encryption key. Then, decrypts the content and play the content according to the usage rules in the license.

5) In revocation scheme, the public authority delegates the key server in the cloud to perform the DFA revocation and user revocation. The DFA revocation will revoke a user's one or more DFAs that he has possessed, which will not influence other users' DFA. However, the user revocation will revoke all of a user's

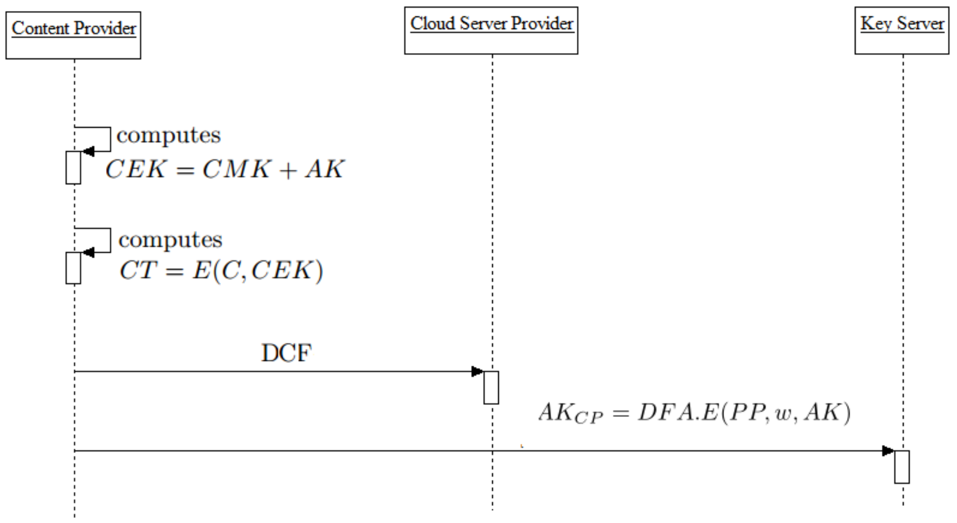


Figure 2: The sequence diagram of content encryption steps

DFA. The revocation scheme operates as follows: In DFA revocation, whenever a user's DFA revocation event is triggered and when the user intends to access the encrypted content, the public authority will inform the key server in the cloud to verify the user's DFA in advance. If the user's DFA cannot satisfy the string w of the encrypted content, the key server refuses to re-encrypt the assistant key for the user. Thus, the user cannot access the content.

6) In user revocation, whenever there is a revoked user intending to access the encrypted contents, the public authority will inform the key server in the cloud to refuse to re-encrypt the assistant key for the user. Hence, immediately after the revocation request is made, our scheme realizes the DFA and user revocations, and in both cases the unauthorized user will not get the licensee.

3.4 Concrete Construction

In this section, we will explain a detailed construction for the proposed scheme as follows:

3.4.1 System Setup and Key Generation

System setup. In this phase, The public authority runs the *Setup* algorithm to generate the public parameters PP and a master key MSK as the following:

$Setup(1^n, \Sigma)$ The setup algorithm selects random group elements $g, g_0, z, h_0 \in \mathbb{G}_{p_1}$ and randomly chooses an exponents $\alpha, k, a, b, \alpha_{End}, \alpha_{Start} \in \mathbb{Z}_N^*$. Then set $H_{Start} = g^{\alpha_{Start}}, H_{End} = g^{\alpha_{End}}$ and $H_k = g^k$. In addition, $\forall\sigma \in \Sigma$ it chooses random $\alpha_\sigma \in \mathbb{Z}_N^*$ and set $H_\sigma = g^{\alpha_\sigma}$. After that it choose a one-time symmetric encryption scheme $Sym = (sym.Enc, sym.Dec)$, a one-time signature scheme Ots and two target collision resistant (TCR) hash functions namely H_1 and H_2, where $H_1 : \mathbb{G}_T \longrightarrow \mathbb{Z}_N^*$ and $H_2 : \mathbb{G}_T \longrightarrow \{0,1\}^{poly(n)}$. Finally, the public authority publishes the public parameters $PP = \{e(g,g)^\alpha, g, g^{ab}, g_0, z, h_0, H_{Start}, H_{End}, H_k, \forall_{\sigma\in\Sigma} H_\sigma,$ $Sym, Ots, H_1, H_2\}$ along with the description of the group $\mathbb{G}$ and the alphabet Σ, while the $MSK = (g^{-\alpha}, X_3)$ is kept secretly by the public authority. Here, X_3 is a generator of $\mathbb{G}_{p_3}$.

Key generation. The public authority runs the $k.Gen(MSK, M = (Q, \tau, q_0, F))$, the key generation algorithm takes the master key MSK and a DFA description M as input. It outputs a private key SK_M, where Q is a set of states $q_0, ..., q_{|Q|-1}$, τ is a set of transitions, for each transition $t \in T$ is a triple $(x, y, \sigma) \in Q \times Q \times \Sigma$. $q_0 \in Q$ is a start state and $F \subseteq Q$ is a set of accept states. The algorithm chooses random group elements $D_0, D_1, ..., D_{|Q|-1} \in \mathbb{G}_{p_1}$, where D_i is associated with state q_i, for each transition $t \in T$ it randomly selects $r_t \in \mathbb{Z}_N^*$, for all $q_x \in F$ it randomly selects $r_{End_x} \in \mathbb{Z}_N^*$, and selects $u \in \mathbb{Z}_N^*$. It also randomly selects $R_{Start_1}, R_{Start_2}, R_{Start_3}, R_{t,1}, R_{t,2}, R_{t,3}, R_{End_{x,1}},$ $R_{End_{x,2}} \in \mathbb{G}_{p_3}$, it also selects randoms $\epsilon, \Omega \in \mathbb{G}_T$ and random $r_{Start} \in \mathbb{Z}_N^*$. The algorithm computes the private key as follows.

Firstly it computes:

$$K_{Start_1} = D_0 \cdot (H_{Start})^{r_{Start}} \cdot R_{Start_1},$$
$$K_{Start_2} = g^{r_{Start}} \cdot R_{Start_2},$$
$$K_{Start_3} = g^u \cdot R_{Start_3}.$$

Secondly, for each transition $t = (x, y, \sigma) \in \tau$ it computes:

$$K_{t,1} = D_x^{-1} \cdot z^{r_t} \cdot R_{t,1},$$
$$K_{t,2} = g^{r_t} \cdot R_{t,2},$$
$$K_{t,3} = D_y \cdot (H_\sigma)^{r_t} \cdot R_{t,3}$$

Thirdly, for all $q_x \in F$ the algorithm sets:

$$K_{End_{x,1}} = g^{-\alpha} \cdot D_x \cdot (H_{End} \cdot g^{ab})^{r_{End_x}} \cdot g^{ku} \cdot R_{End_{x,1}},$$
$$K_{End_{x,2}} = g^{r_{End_x}} \cdot R_{End_{x,2}}.$$

Finally, the key public authority generates the SK_M and sends it to the authorized user in secure channel. SK_M is denoted as: $SK_M = (M, \epsilon, \Omega, DK)$, where $DK = (DK_1 = {K_{Start_1}}^{H_1(\epsilon)}, DK_2 = {K_{Start_2}}^{H_1(\epsilon)}, DK_3 = {K_{Start_3}}^{H_1(\epsilon)}, \forall t \in \tau(DK_{t,1} = {K_{t,1}}^{H_1(\epsilon)}, DK_{t,2} = {K_{t,2}}^{H_1(\epsilon)}, DK_{t,3} = {K_{t,3}}^{H_1(\epsilon)}), \forall q_x \in F(DK_{End_{x,1}} = {K_{End_{x,1}}}^{H_1(\epsilon)}, DK_{End_{x,2}} = {K_{End_{x,2}}}^{H_1(\epsilon)}))$.

The public authority also runs the re-encryption algorithm $ReKeyGen(SK_M, w)$ to generate the re-encryption key $Rk_{M\to w}$ for authorized user and sends it to the key server in secure channel as follows:

Firstly, the public authority selects random $\beta_r \in \mathbb{Z}_N^*$ for all $q_x \in F$. Then computes $Rk_1 = K_{start_1}^{H_1(\Omega)}$, $Rk_2 = K_{start_2}^{H_1(\Omega)}$, $Rk_3 = K_{start_3}^{H_1(\Omega)}$, for all $t \in \tau(RK_{t,1} = K_{t,1}^{H_1(\Omega)}, RK_{t,2} = K_{t,2}^{H_1(\Omega)}, RK_{t,3} = K_{t,3}^{H_1(\Omega)})$, for all $q_x \in F(Rk_{End_{x,1}} = K_{End_{x,1}}^{H_1(\Omega)} \cdot H_{End}^{\beta_r}, Rk_{End_{x,2}} = K_{End_{x,2}}^{H_1(\Omega)} \cdot g^{\beta_r})$.

Finally, the $Rk_{M\to w} = (M, Rk_1, Rk_2, Rk_3, \forall t \in \tau(RK_{t,1}, RK_{t,2}, RK_{t,3}), \ \forall \ q_x \in F(Rk_{End_{x,1}}, Rk_{End_{x,2}}))$.

3.4.2 Content Packaging and Encryption

The content provider process in this phase are represented as follows:

- Firstly, the content provider computes CEK such as $CEK = CMK + AK$. Then using symmetric encryption algorithm encrypts the content such as $CT = E_{sym}(C, CEK)$.

- Secondly, the content provider encrypts the CMK using *sym.Enc* encryption algorithm and obtains the CMK_M as follows: $CMK_M = sym.Enc(H_2(CID), CMK)$, where the CID is the content identity. Then outsources the DCF to the CSP.

- Finally, the content provider encrypts the AK using $DFA.E(PP, w, AK)$ encryption algorithm and obtains the AK_{CP} as follows:

  The content provider randomly selects $\lambda_0, \lambda_1, ..., \lambda_l \in \mathbb{Z}_N^*$, run $(ssk, svk) \longleftarrow KeyGen(1^n)$ and computes AK_{CP} as

  First set: $C_{AK} = AK \cdot e(g,g)^{\alpha\cdot\lambda_l}$, $C_{Start_1} = C_{0,1} = g^{\lambda_0}$, $C_{Start_2} = (H_{Start})^{\lambda_0}$, $C_{Start_3} = (g_0^{svk} h_0)^{\lambda_0}$, for $i = 1$ to l, set: $C_{i,1} = g^{\lambda_i}, C_{i,2} = (h_{wi})^{\lambda_i} \cdot z^{\lambda_i - 1}$, finally, set:
  $C_{End_1} = C_{l,1} = g^{\lambda_l}$, $C_{End_2} = (H_{End} \cdot g^{ab})^{\lambda_l}$, $C_{End_3} = (H_k)^{\lambda_l}$,
  $C_{End_4} = Sign(ssk, (w, C_{AK}, C_{Start_1}, C_{Start_2}, C_{Start_3}, (C_{1,1}, C_{1,2}), ..., (C_{End_1}, C_{l,2}), C_{End_2}, C_{End_3}))$.

  The ciphertext AK_{CP} is
  $AK_{CP} = (svk, w, C_{AK}, C_{Start_1}, C_{Start_2}, C_{Start_3}, (C_{1,1}, C_{1,2}), ..., (C_{l,1}, C_{l,2}), C_{End_2}, C_{End_3}, C_{End_4})$.

3.4.3 License Acquisition

Upon receiving the user's license acquisition request, the key server computes the re-encrypted AK_{CP} with $Rk_{M\to w'}$ using the re-encryption algorithm $ReEnc(Rk_{M\to w'}, AK_{CP})$. The process in this phase represented as follows:

- The key server checks $verify(svk, (w, C_{AK}, C_{Start_1}, C_{Start_2}, C_{Start_3}, (C_{1,1}, C_{1,2}), ..., (C_{End_1}, C_{l,2}), C_{End_2}, C_{End_3})) = 1$ and $e(C_{Start_1}, g_0^{svk} h_0) = e(g, C_{Start_3})$, outputs "True" if valid and "False" otherwise. If the verification is hold then proceed.

- The string $w = (w_1, ..., w_l)$ is associated with the AK_{CP} and the DFA $M = (Q, \tau, q_0, F)$ is associated with the user's re-encryption key $Rk_{M\to w'}$ where $ACCEPT(M, w)$. There must exist a sequence of $l+1$ states $\mu_0, \mu_1, ..., \mu_l$ and l transitions $t_1, ..., t_l$ where $\mu_0 = q_0$ and $\mu_l \in F$, we have $t_i = (\mu_{i-1}, \mu_i, w_i) \in \tau$. The key server re-encrypts AK_{CP} as follows:

  – First computes:

  $$\begin{aligned}\phi_0 &= e(C_{Start_1}, Rk_1) \cdot e(C_{Start_2}, Rk_2)^{-1} \\ &= e(g, D_0)^{\lambda_0 \cdot H_1(\Omega)}.\end{aligned}$$

  – For $i = 1$ to l, computes:

  $$\begin{aligned}\phi_i &= \phi_{i-1} \cdot e(C_{(i-1),1}, Rk_{t_i,1}) \\ &\quad \cdot e(C_{i,2}, Rk_{t_i,2})^{-1} \cdot e(C_{i,1}, Rk_{t_i,3}) \\ &= e(g, D_{\mu_i})^{\lambda_i \cdot H_1(\Omega)}.\end{aligned}$$

  whenever M accepts w, we have that $\mu_l = q_x$ for some $q_x \in F$ and $\phi_l = e(g, D_x)^{\lambda_l \cdot H_1(\Omega)}$.

  – Then sets:

  $$\begin{aligned}\phi_{End} &= \phi_l \cdot e(C_{End_{x,1}}, Rk_{End_{x,1}})^{-1} \\ &\quad \cdot e(C_{End_{x,2}}, Rk_{End_{x,2}}) \cdot e(C_{End_{x,3}}, Rk_3) \\ &= e(g,g)^{\alpha\cdot\lambda_l\cdot H_1(\Omega)}.\end{aligned}$$

  – The key server selects random $\gamma \in \mathbb{G}_T$ and sets $\pi_1 = sym.Enc(H_2(\gamma), A)$ and $\pi_2 = DFA.E(PP, w', \gamma)$, where $A = (AK_{CP} \parallel \phi_{End})$. Finally, the key server sends $AK_{CP}^R = (\pi_1, \pi_2)$ to the license server.

3.4.4 Content Consumption

The user recovers the CMK from the ciphertext CMK_M as follows:

$$CMK = sym.Dec(H_2(CID), CMK_M) \qquad (1)$$

The user whose the DFA associated with the his/her secret key accepts the string w will recover the AK as follows:

- Firstly, computes γ as follows: The user sends his partial decryption key DK to the cloud service provider for partial decryption, the cloud service provider works the following:

  If $verify(svk, (w, C_{AK},\ C_{Start_1},\ C_{Start_2},\ C_{Start_3}, (C_{1,1},\ C_{1,2}),\ \cdots,\ (C_{End_1},\ C_{l,2}),\ C_{End_2},\ C_{End_3})) = 1$ and $e(C_{Start_1}, g_0^{svk} h_0) = e(g, C_{Start_3})$, outputs "True" if valid and "False" otherwise. If the verification hold, proceed.

  Then cloud service provider computes $AK_{CP}{}'$ as follows:

  - $\theta_0 = e(C_{Start_1}, DK_1) \cdot e(C_{Start_2}, DK_2)^{-1} = e(g, D_0)^{\lambda_0 \cdot H_1(\epsilon)}$.
  - For $i = 1$ to l, compute:

  $$\begin{aligned}\theta_i &= \theta_{i-1} \cdot e(C_{(i-1),1}, DK_{t_i,1}) \\ &\quad \cdot e(C_{i,2}, DK_{t_i,2})^{-1} \cdot e(C_{i,1}, DK_{t_i,3}) \\ &= e(g, D_{\mu_i})^{\lambda_i \cdot H_1(\epsilon)},\end{aligned}$$

    whenever M accepts w, we have that $\mu_l = q_x$ for some $q_x \in F$ and $\theta_l = e(g, D_x)^{\lambda_l \cdot H_1(\epsilon)}$.
  - Finally compute:

  $$\begin{aligned}\theta_{End} &= \theta_l \cdot e(C_{End_{x,1}}, DK_{End_{x,1}})^{-1} \\ &\quad \cdot e(C_{End_{x,2}}, DK_{End_{x,2}}) \\ &\quad \cdot e(C_{End_{x,3}}, DK_3) \\ &= e(g,g)^{\alpha \cdot \lambda_l \cdot H_1(\epsilon)}\end{aligned}$$

    and sends the message $AK_{CP}{}' = \theta_{End}$ to the user within his licence. The user then can retrieve γ as follows:

  $$\gamma = C_\gamma / \{AK_{CP}{}'\}^{H_1(\epsilon)^{-1}}$$

    We note that $C_\gamma = \gamma \cdot e(g,g)^{\alpha \cdot \lambda_l}$, since it encrypted using the $DFA.E(PP, w, m)$ encryption algorithm.

- Secondly, the user computes A as follows:

  $$A \longleftarrow sem.Dec(H2(\gamma), \pi_1)$$

  Where $A = (AK_{CP} \parallel \phi_{End})$.

- Thirdly, computes Key as follows:

  $$Key = \phi_{End}^{H_1(\Omega)^{-1}}$$

- Finally, outputs the AK as follows:

  $$AK = C_{AK}/Key$$

Then if the user's UR are effective, the user can computes the CEK such as:

$$CEK = CMK + AK$$

Finally the user decrypts the encrypted content and plays the content according to the RE in the license.

$$C = D(CEK, CT)$$

4 Analysis

4.1 Correctness

The correctness of our scheme is extremely straightforward. After downloading the DFC header from the content service provider, the user extracts the CMK_M from the DFC header. The CMK_M is a ciphertext of using a *sym.Enc* encryption algorithm. So, the user recovers the CMK from the ciphertext CMK_M as follows:

$$CMK = sym.Dec(H_2(CID), CMK_M)$$

In the license acquisition phase, the key server re-encrypts the AK_{CP} to the AK_{CP}^R without disclosing the AK. Moreover, AK_{CP} is converted to a re-encrypted ciphertext AK_{CP}^R under w. Thus, if the user has been granted the right DFA, the user should own the corresponding secret key for decryption (SK_M). Hence, only the user can recover the plain text of AK with the private key SK_M as follows:

$$\begin{aligned}AK &= C_{AK}/Key \\ &= AK \cdot e(g,g)^{\alpha \cdot \lambda_l} / \phi_{End}^{H_1(\Omega)^{-1}} \\ &= AK \cdot e(g,g)^{\alpha \cdot \lambda_l} / \{e(g,g)^{\alpha \cdot \lambda_l \cdot H_1(\Omega)}\}^{H_1(\Omega)^{-1}} \\ &= AK\end{aligned}$$

Therefore, with the possibility of recovering the CMK and AK, the user can surely decrypt the content.

4.2 Security

Our proposed scheme relies on K. Liang et al.'s scheme [26]. It has been proven to be secure in the standard model, and it seems suitable for DRM system. Therefore, we focus on the following theorems.

Theorem 1. *It is only feasible for an authorized user to access the contents.*

Proof. As we presented, the user who has a matching of the DFA and effective usage rights only can decrypt the content. Hence, our scheme provides access of contents only for authorized users. The proposed scheme ensures

Table 2: Efficiency comparison

Scheme	**Fine-grained access control**	**Revocation scheme**	**Privacy preserving**	**Access policy size input**	**Complexity of content decryption**
ref[23]	No	No	Yes	None	$8T_a + T_r$
ref[17]	Yes	No	N/A	Limited	$T_b + T_{dec}$
ref[4]	Yes	Yes	Yes	Limited	$T_b + T_a + T_{dec}$
ref[5]	Yes	Yes	Yes	Limited	$T_b + T_{dec} + 2T_{exp}$
our scheme	Yes	Yes	Yes	Unlimited	$3T_{dec} + 2T_{exp}$

Table 3: Efficiency comparison against Huang's scheme

	Operations								Total running time m/s
	Pairing		Pairing-based scalar multiplication		Symmetric decryption		Exponential operation		
	Number	Running time	Number	Running time	Number	Running time	Number	Running time	
Ref. [5]	3	61.2	4	25.52	1	3.04	2	21.28	111.04
Our	0	0	0	0	3	9.12	2	21.28	30.4

confidentiality of the content from the following four aspects. Firstly, if the unauthorized user can retrieve the desired value $e(g,g)^{\alpha \cdot \lambda_l}$, which is required for the decryption into two cases, the DFA revocation and the user revocation, he also cannot recover the AK. In fact, if the user's DFA cannot satisfy the string w of the encrypted content, the key server will refuse to re-encrypt the assistant key for the user. On the other hand, when a user is revoked, he cannot recover AK without the re-encrypt the assistant key part in the user revocation case. Therefore, unauthorized user cannot recover the AK. Secondly, the license server cannot get the content master key. Thirdly, the key server cannot get the plain assistant key. Fourthly, the curious CSP cannot read the contents without the content encryption key since any of private keys is not given to the CSP from the content provider in our scheme. On the other hand, even if CSP colludes with some user transform the ciphertext AK_{CP} to $AK_{CP}^{'}$ and obtain $AK_{CP}^{'}$ using the user's partial decryption key DK, he still cannot recover the AK, because he does not know the secret values ϵ and Ω. Therefore, neither a curious CSP nor unauthorized users or license server in the cloud can read the contents. □

Theorem 2. *It is infeasible for an illegal user to get the license from the malicious employees of license server.*

Proof. In the license acquisition phase, license server only can receive the re-encrypted ciphertext AK_{CP}^{R} . Therefore, malicious employees of license server cannot issue license to illegal user without the full content encryption key. □

Theorem 3. *It is infeasible for an attacker to replay license acquisition request and key acquisition request.*

Proof. In our scheme, the service provider sends license acquisition request $LSQ = \{CID \parallel UR \parallel T \parallel \sigma_{LS} \parallel AK_{CP}^{'}\}$ to the license server. Upon receiving the user's license acquisition request, the license server checks the signature σ_{LS} and T. If the adversary E can modify it to $LSQ^{'} = \{CID \parallel UR \parallel T^{'} \parallel \sigma_{LS} \parallel AK_{CP}^{'}\}$ and send $LSQ^{'}$ to the license server, the license server concludes that $T^{'} \neq T$, and rejects the request. Thus, the replaying license acquisition request is impossible.

In the same way, the license server sends key acquisition request $\{CID \parallel T \parallel \sigma_{KS}\}$ to the key server. Upon receiving the user's key acquisition request, the key server checks the signature σ_{KS} and T. If the adversary E can modify it to $\{CID \parallel T^{'} \parallel \sigma_{KS}\}$ and send it to key server, the key server concludes that $T^{'} \neq T$, and rejects the request. Hence, the replaying key acquisition request is infeasible. □

Theorem 4. *Our key construction mechanism is considered to be secure due to the collusion-resistant for the users.*

Proof. Using symmetric encryption algorithm, the CMK is protected and distributed within the encrypted content. On the other hands, the AK is protected using the DFA-based FPRE encryption algorithm and then stored in the key server. The users who fulfill the string w can get the plain text of the content master key, and then obtain assistant key from key server when they intend to play the contents. We showed that the CEK is not available for more than one user to collude or access. The conspiring users can retrieve the desired values $e(g,g)^{\alpha\cdot\lambda_l\cdot H_1(\epsilon)}$ and $e(g,g)^{\alpha\cdot\lambda_l\cdot H_1(\Omega)}$ in order to recover the required value $e(g,g)^{\alpha\cdot\lambda_l}$ for decryption operation. However, the two values ϵ and Ω are random and unique exponents for each user, which renders the combination of information in different users' secret keys meaningless. Even if the users obtain the CMK, they cannot compute the CEK, since the CEK is computed by adding the CMK and AK. In this sense, such collusion attack can be precluded in our construction scheme. □

4.3 Privacy Preserving

In our scheme, an anonymous user directly communicates with the cloud service provider and key server, which prevents the other parties from getting any user's personal information, for example, which software is bought and who bought the software. In the key generation phase, an anonymous user register to the CSP and then get it's public/private key issued by key server and secret keys SK_M issued by public authority. In the content decryption phase, the user acquires AK from the key server without giving out any personal information. As a result, the user's privacy is maintained.

4.4 Performance Analysis

For convenience, in this section we define the following notations: T_H (the time complexity of one-way hash function); T_e (the time complexity of pairing operation); T_r (the time complexity of proxy re-encryption); T_{mul} (the time complexity of pairing-based scalar multiplication); T_{exp} (the time complexity of exponential operation); T_b (represents the attribute-based encryption); T_{dec} (the time complexity of symmetric decryption); T_a (represents the asymmetric encryption).

We compare our scheme with existing DRM schemes, in terms of access control, revocation scheme, the access policy size input and privacy preserving. The results are given in Table 2. It is easy to find that Petrlic et al.'s scheme [23] has higher computational cost than our scheme since their scheme uses eight times asymmetric encryption operations at the user side. Whereas, it does not provide a revocation method. For the other related attribute-based encryption DRM schemes [4, 5, 17], the encryption costs are almost the same, which increases linearly with the number of attributes used in the data encryption. However, the decryption cost for the user in our scheme is much less than these schemes, which just includes two modular exponentiation operations and three symmetric decryption operations. Therefore, our scheme has much better efficiency.

In Table 3 the efficiency comparison of our scheme against Huang et al. [5] scheme which has concrete construction is presented. This comparison is prepared based on experimental results in [6, 7], for various cryptographic operations using MIRACLE [18] in PIV 3 GHZ platform processor with memory 512 MB and the Windows XP operating system. From these experimental results, the relative running time of one pairing operation T_e is 20.04 m/s, one-way hash function T_h is 3.04 m/s, pairing-based scalar multiplication T_{mult} is 6.38 m/s and exponential operation T_{exp} is 10.64 m/s, where the symmetric key encryption and decryption running times are very close to hash function running time [26, 29]. Hence, we adopted the same running time of one-way hash function for both encryption T_{enc} and decryption T_{dec}.

Let N_u, N_c and N_a denote the number of users, contents, and attributes respectively. In ABE schemes, the computational cost increases linearly either to $N_u \cdot N_a$ or $N_c \cdot N_a$, but never linearly to the product of the three $N_u \cdot N_c \cdot N_a$ [2, 3]. In our scheme, the computational costs in the content consumption phase at the user side is $3T_{dec} + 2T_{exp}$.

As indicated in Table 3, the computational cost of Huang et al.'s scheme is increasingly higher. Furthermore, the decryption cost in this scheme increases linearly with the number of attributes. Moreover, it requires three times bilinear pairing operation. However, the time consumed in pairing operation is more than other operations over elliptic curve group. Finally, Figure 3 shows the efficiency comparison of our scheme versus Huang et al. based on running time for each operation.

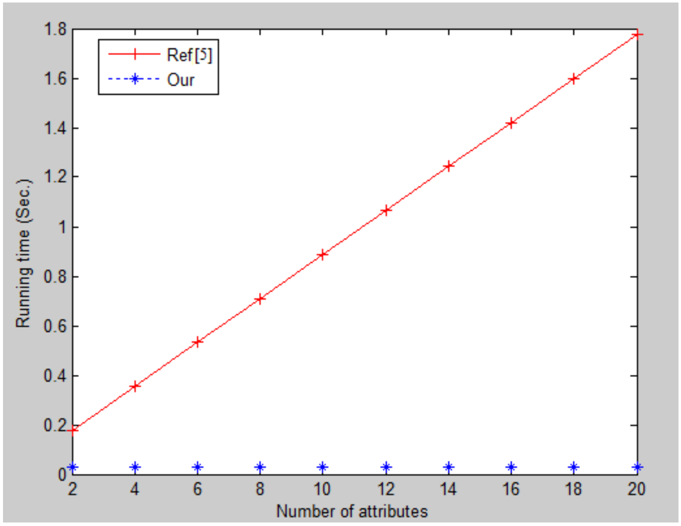


Figure 3: The efficiency comparison against Huang's scheme

5 Conclusions

Based on DFA-based FPRE scheme, we proposed a secure, efficient, and fine-grained access control system for DRM system. Furthermore, we put forward a mechanism for distributing licenses in a flexible and secure manner. In our scheme, the user who has a DFA associated with his/her secret key accepts the string associated with the ciphertext and has effective usage rights can quite efficiently access the encrypted content with the help of the cloud service provider, and the revocation is both flexible and fine-grained. Moreover, the revocation task can be made immediately without disclosing the ciphertext. Finally, comparing with other DRM schemes in cloud computing, it is safe to draw the conclusion that our present work could be considered a secure and high efficient work for DRM system.

Acknowledgments

The author would like to acknowledge National Natural Science Foundation of China under Grant (No. 61003230, 61370026 and 61202445), the Fundamental Research Funds for the Central Universities under Grant (No. ZYGX2013J073 and ZYGX2012J067).

References

[1] H. Abdalla, X. Hu, A. Wahaballa, P. Avornyo and Q. Zhiguang, "Anonymous pairing-free and certificateless key exchange protocol for DRM system," *International Journal of Network Security*, vol. 18, no. 2, pp. 235–243, 2016.

[2] J. Camenisch, M. Dubovitskaya, G. Neven, G. M. Zaverucha, "Oblivious transfer with hidden access control policies" in *Public Key Cryptography*, LNCS 6571, pp. 192–209, Springer, 2011.

[3] J. Camenisch, M. Dubovitskaya, R. R. Enderlein and G. Neven, "Oblivious transfer with hidden access control from attribute-based encryption," in *Conference on Security and Cryptography for Networks,* PP. 1–32, 2012.

[4] Q. Huang, Z. Ma, J. Fu, X. Niu and Y. Yang, "Attribute based DRM scheme with efficient revocation in cloud computing," *Journal of Computers,* vol. 8, no. 11, pp. 2776–2781, 2013.

[5] Q. Huang, Z. Ma, Y. Yang, X. Niu and J. Fu, "Attribute based DRM scheme with dynamic usage control in cloud computing," *Communications,* vol. 11, no. 4, pp. 50–63, 2014.

[6] D. He, J. Chen and R. Zhang, "An efficient identity-based blind signature scheme without bilinear pairings," *Computer Electrical Engineering,* vol. 37, no. 4, pp. 444–450, 2011.

[7] D. He and J. Chen, "An efficient certificateless designated verifier signature scheme," *International Arab Journal of Information Technology,* vol. 10, no. 4, pp. 389–396, 2013.

[8] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel and W. Jonker, "Mediated ciphertext-policy attribute-based encryption and its application," in *Information Security Applications,* pp. 309–323, Springer Berlin Heidelberg, 2009.

[9] A. Lewko, T. Okamoto, A. Sahai, K. Takashima and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *Advances in Cryptology (Eurocrypt'10),* pp. 62–91, Springer Berlin Heidelberg, 2010.

[10] A. Lewko and B. Waters, "New proof methods for attribute-based encryption: Achieving full security through selective techniques," in *Advances in Cryptology (Crypto'12)*, pp. 180–198, Springer Berlin Heidelberg, 2012.

[11] X. Liang, Z. Cao, H. Lin and J. Shao, "Attribute based proxy re-encryption with delegating capabilities," in *ACM Proceedings of the 4th International Symposium on Information, Computer, and Communications Security,* pp. 276–286, 2009.

[12] S. Luo, J. Hu and Z. Chen, "Ciphertext policy attribute-based proxy re-encryption," in *Information and Communications Security,* pp. 401–415, Springer Berlin Heidelberg, 2010.

[13] K. Liang, M. Au, J. Liu, W. Susilo, D. Wong, G. Yang, P. Tran and Q. Xie, "A DFA-based functional proxy re-encryption scheme for secure public cloud data sharing", *IEEE Transactions on Information Forensics and Security,* vol. 9, no. 10, pp. 1667–1680, 2014.

[14] A. Lewko, T. Okamoto, A. Sahai, K. Takashima and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *Advances in Cryptology (Eurocrypt'10),* pp. 62–91, Springer Berlin Heidelberg, 2010.

[15] T. Mizuno and H. Doi, "Hybrid proxy re-encryption scheme for attribute-based encryption," in *Information Security and Cryptology,* pp. 288–302, Springer Berlin Heidelberg, 2010.

[16] Z. Ma, K. Fan, M. Chen, Y. Yang and X. Niu, "Trusted digital rights management protocol supporting for time and space constraint," *Journal on Communications*, vol. 29, no. 10, pp. 153–164, 2008.

[17] S. Muller, S. Katzenbeisser, "A new DRM architecture with strong enforcement," in *Proceedings of the 5th International Conference on Availability, Reliability, and Security,* pp. 397–403, 2010.

[18] MIRACL, Multiprecision Integer and Rational Arithmetic C/C++ Library, 2016. (`http://indigo.ie/mscott/`)

[19] T. Okamoto and K. Takashima, "Fully secure functional encryption with general relations from the decisional linear assumption," in *Advances in Cryptology (Crypto'10),* pp. 191–208, Springer Berlin Heidelberg, 2010.

[20] T. Okamoto and K. Takashima, "Fully secure unbounded inner-product and attribute-based encryption," in *Advances in Cryptology (Asiacrypt'12)*, pp. 349–366, Springer Berlin Heidelberg, 2012.

[21] R. Petrlic and C. Sorge, "Privacy-preserving DRM for cloud computing," in *IEEE 26th International*

Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 1286–1291, 2012.

[22] R. Perlman, C. Kaufman, R. Perlner, "Privacy-preserving DRM," in *Proceedings of the 9th Symposium on Identity and Trust on the Internet,* pp. 69–83, 2010.

[23] P. Ronald, "Proxy re-encryption in a privacy-preserving cloud computing DRM scheme," in *Cyberspace Safety and Security,* pp. 194–211, Springer Berlin Heidelberg, 2012.

[24] M. Ruiz, D. M. L and J. Pedraza, "Privacy risks in cloud computing," in *Intelligent Agents in Data-intensive Computing,* Springer International Publishing, pp. 163–192, 2016.

[25] S. C. Ramanna, "DFA-based functional encryption: Adaptive security from dual system encryption," *IACR Cryptology ePrint Archive*, vol. 2013, pp. 638, 2013.

[26] S. Ramesh, V. M. Bhaskaran, "An improved remote user authentication scheme with elliptic curve cryptography and smart card without using bilinear pairings," *International Journal of Engineering and Technology (IJET),* vol. 5, no. 6, 2014.

[27] A. Wahaballa, Z. Qin, H. Xiong, Z. Qin and M. Ramadan, "A taxonomy of secure electronic english auction protocols," *International Journal of Computers and Applications,* vol. 37, no. 1, pp. 28–36, 2015.

[28] B. Waters, "Functional encryption for regular languages," in *Advances in Cryptology (Crypto'12),* LNCS 7789, pp. 218–235, Springer, 2012.

[29] D. Wang, D. He, P. Wang, et al., "Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment," *IEEE Transactions on Dependable and Secure Computing,* vol. 12, no. 4, pp. 428–442, 2014.

[30] S. Yu, C. Wang, K. Ren and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *IEEE International Conference on Computer Communications,* pp. 1–9, 2010.

[31] Z. Zhang, Q. Pei, J. Ma, and L. Yang, "Establishing multi-party trust architecture for DRM by using game-theoretic analysis of security policies," *Chinese Journal of Electronics*, vol. 18, no. 3, pp. 519–524, 2009.

Hisham Abdalla is a doctorial student at University of Electronic Science and Technology of China (UESTC). He received his M.Sc. degree from UESTC and BE degree in computer engineering from Karary University in 2006. His research interests include cloud computing security, cryptography and digital right management.

Xiong Hu is an associate professor in the School of Information and Software Engineering, UESTC. He received his Ph.D. degree from UESTC in 2009. His research interests include: information security and cryptography.

Abubaker Wahaballa received his PhD. degree from University of Electronic Science and Technology of China. His current research interests include information security, cryptography, steganography and DevOps.

Ahmed Abdalla was born in 1982. He received the B.S. Degree in Electrical Engineering from Karary University in 2005, Khartoum Sudan and the M.Sc. in Electronic Engineering, Information and signal processing form University of Electronic Science and Technology of China in 2013, Chengdu, China. He is currently working toward the PhD. Degree Electronic Engineering from University of Electronic Science and Technology of China. His current research interests include radar counter countermeasure and radar signal processing.

Mohammed Ramadan received the B.S. Degree in Communication Engineering from Karary University in 2007, Khartoum Sudan and the M.S. Degree in M.Sc. in Computer Engineering, Information Security form University of Electronic Science and Technology of China in 2013, Chengdu, China. He is currently working toward the PhD. degree in Information Security, Mobile Communication Security from University of Electronic Science and Technology of China. His current research interests include Wireless and Mobile Communications security (LTE security).

Qin Zhiguang is a professor at University of Electronic Science and Technology of China (UESTC). Research interest: network security, social network. He has published more than 100 papers on international journals and conference among which more than 50 are indexed by SCI and EI. He has been principal investor of 2 NSF key projects, 2 sub-topics of national major projects and 6 national 863 projects.

An Improved Privacy Protection Security Protocol Based on NFC

Jie Ling[1], Ying Wang[1], Weifeng Chen[2]
(Corresponding author: Ying Wang)

Faculty of Computer, Guangdong University of Technology, Guangzhou 510006, China[1]
Computer Science & Information Systems, California University of Pennsylvania California, PA 15419, USA[2]
(Email: 526047587@qq.com)

(Received Sep. 15, 2015; revised and accepted Dec. 7 & Dec. 30, 2015)

Abstract

An improved NFC-based privacy protection protocol is proposed to protect user's privacy in NFC application. In this paper, user's privacy is protected by Chebyshev-map and certificateless public key cryptography in the protocol. As a trusted third party, Trusted Service Manager (TSM) participates in the process of user registration and verifies the identity of the parties if necessary. With high calculation speed, the proposed method can eliminate vulnerabilities of impersonations attacks and save the storage space of NFC devices.

Keywords: Identity authentication, NFC, privacy preserve, pseudonym

1 Introduction

NFC (Near Field Communication) is a short-range wireless communication technology that evolved from RFID which has been studied by scholars [15, 16, 18, 19], it's technology distance is around 4 inches, and it operates in the 13.56 MHz frequency band at a speed of 106 Kbps, 212 Kbps or 424 Kbps. NFC technology has been widely used in smart phones and other consumer electronics devices. The mobile phone with NFC-enabled can be used for mobile payment, e-ticketing, intelligent media browsing and data transmission and exchange, etc [5, 11]. It is vulnerable to eavesdropping, data tampering, data corruption, cloning and phishing attacks, resulting in the leak of user privacy data which is a serious threat to financial information and the user's property security.

How to strengthen and improve the security of NFC has become a hot issue of academic and industrial circles in recent years. A series of NFC security standards have been formulated. They expressly stipulated that key agreement is required for secret communications between users [9, 10]. In the key agreement, both users should exchange their certificates to get the public key of another party. Specifically, the certificate includes the user's personal information. Thus, the attacker can get the user's action and privacy by tracing the public key.

As an important privacy protection method, the pseudonym-based privacy protection methods have been widely used in many applications [6, 8, 12, 17]. In such applications, the user's identity is represented by a pseudonym, which is generated by the third trusted services manager randomly. It means that the user's identity has no relation to the user's real identity. Therefore, the attacker cannot get the user's real identity even if he could get the user's pseudonym. A lot of research have been done to strengthen the security of NFC. In [2], Chi et al. proposed the secure transaction protocol in NFC card emulation mode, but this protocol is no privacy protection. The Reference [20] proposed an NFC mobile trusted anonymous authentication enhanced privacy protection model which realized the anonymous authentication of users. However, this model requires an embedded mobile trusted computing module, which is difficult to implement in practical application. Eun et al. proposed an conditional privacy preserving security protocol for NFC-based applications [4].Eun et al. proposed an improved conditional privacy preserving security protocol for NFC-based applications [3]. However it updates pseudonym without the communication with the TSM which can be used only to keep track of the message constructor. So it has high computational efficiency. According to reveal that Eun et al.'s protocol could not withstand impersonation attack, the Reference [7] proposed an improved method, in which the TSM is responsible for generating pseudonym set and verifying the identity of users. The improved method can resist impersonation but results in additional space to store pseudonym set and low calculation efficiency.

This paper proposed an improved privacy protection security protocol based on NFC. Users need to register at the TSM and get the security information which will be involved in the process of key agreement to ensure the security and privacy of the protocol and mobile devices do not need more storage space. This protocol can withstand impersonation attack, improved the computing efficiency,

Table 1: Notation

Notation	Description
N_A	Nonce of A
ID_A	Random ID of user A
G	Elliptic curve base point
d_A	Private key of user A
Q_A	Public key of user$A$,Q_A=d_AG
Q_s	Public key of TSM
Z	shared secret value taking x-coordinate from P
r_A	Random integer generated by user A
SSK	Shared secret key
KDF	Key derivation function
$MacTag_A$	Key verification tag received from A
$Enc(k,m)$	Encrypt m with k
$Sig(k,m)$	Signature on m with k

and reduce the storage of NFC devices.

The rest of the paper is organized as follows. Section 2 describes background knowledge related to proposed protocol. In Section 3, Eun et al.'s and Debiao et al.'s security protocol are provided. In Section 4, the proposed secure protocol based NFC is introduced. Security and performance analysis results are given in Section 5. Section 6 draws conclusions.

2 Background Knowledge

2.1 Chebyshev-map

Chebyshev-map [13] $T_n(x) : [-1,1] \to [-1,1]$ is mapping defined by n-degree Chebyshev polynomials $T_n(x) = \cos(n \times \arccos(x)), x \in [-1,1]$. The definition of recursive: $T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), n \geq 2, x \in [-\infty, +\infty]$ and $T_0(x)$=1,$T_1(x)$=x.

Chebyshev polynomials satisfy the semigroup properties:$T_r(T_s(x)) = T_{r\times s}(x)$, and also meet the commutative:$T_r(T_s(x)) = T_s(T_r(x))$.

2.2 Pseudonym Composition

Pseudonym represents ID that changes randomly. The pseudonym are composed of the user's public key Q_A^i, the user's private key d_A^i,the TSM's identity and the TSM's signatureS_{TSM}^i.Pseudonym composition [8].

$$\begin{aligned} S_{TSM}^i &= Sig(d_{TSM}, Q_A^i \parallel Enc(Q_A^i, d_A^i) \parallel ID_{TSM}) \\ PN_A^i &= \{Q_A^i \parallel Enc(Q_A^i, d_A^i) \parallel S_{TSM}^i\}. \end{aligned}$$

Pseudonym can guarantee user anonymity, protect personal privacy. Usually pseudonym-based privacy protection method uses pseudonym set generated by TSM. TSM need to stores pseudonyms and the actual ID of users to reveal the anonymity in case of a problem. However, the method using the pseudonym requires additional costs for storage and communication. In order to improve the insufficiency of the user need additional storage space to save pseudonym set in NFC devices, some researchers are studying the methods of generating pseudonym without the help of TSM.

2.3 Notations

The notations used in the paper as shown in Table 1.

3 Related Work

3.1 Eun et al.'s Privacy Preserving Protocol

In order to protect the user's privacy, Eun et al [3] proposed a self-updateable pseudonym based method which can update pseudonym without the need to communicate with TSM. TSM only reveals the true identity of users in needed, and the communication and computation efficiency is high. The protocol process shown in Figure 1 and as follows:

1) A generates a nonce N_A and a random number r_A, Then, A computes $Q_A^{'} = r_A Q_A$, $Q_A^{''} = r_A d_A Q_s + Q_A^{'}$ and sends the message $m_1 = \{Q_A^{'} \parallel N_A \parallel Q_A^{''}\}$ to B.

2) Upon receiving the message m_1, B generates a nonce N_B and a random number r_B. B computes $Q_B^{'} = r_B Q_B$, $Q_B^{''} = r_B d_B Q_s + Q_B$ and sends the message $m_2 = \{Q_B^{'} \parallel N_B \parallel Q_B^{''}\}$ to A.

3) Upon receiving the message m_2, A computes $P = r_A d_A Q_B^{'}$, $SSK = KDF(N_A, N_B, ID_A, Z)$, $MacTag_A = f(SSK, ID_A, Q_A, Q_B)$ and sends message $m_3 = \{MacTag_A\}$.

4) Upon receiving the message m_3, B computes $P = r_B d_B Q_A^{'}$, $SSK = KDF(N_A, N_B, ID_A, Z)$,

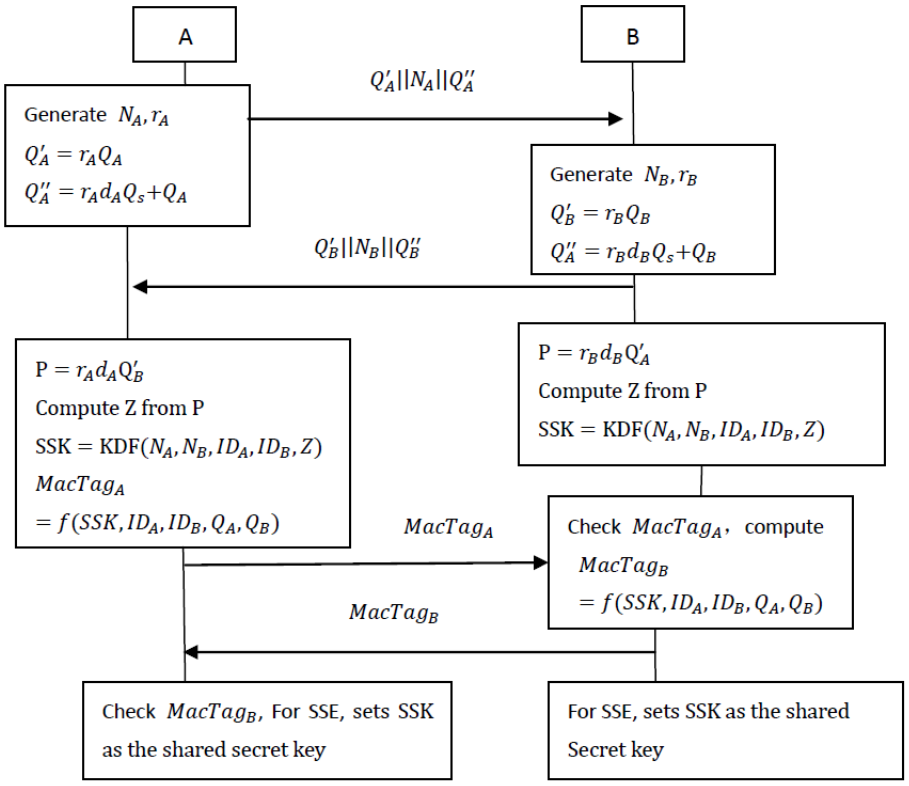

A
B
$Q'_A||N_A||Q''_A$
Generate N_A, r_A
$Q'_A = r_A Q_A$
$Q''_A = r_A d_A Q_s + Q_A$
Generate N_B, r_B
$Q'_B = r_B Q_B$
$Q''_A = r_B d_B Q_s + Q_B$
$Q'_B||N_B||Q''_B$
P = $r_A d_A Q'_B$
Compute Z from P
SSK = KDF(N_A, N_B, ID_A, ID_B, Z)
$MacTag_A = f(SSK, ID_A, ID_B, Q_A, Q_B)$
P = $r_B d_B Q'_A$
Compute Z from P
SSK = KDF(N_A, N_B, ID_A, ID_B, Z)
$MacTag_A$
Check $MacTag_A$, compute $MacTag_B = f(SSK, ID_A, ID_B, Q_A, Q_B)$
$MacTag_B$
Check $MacTag_B$, For SSE, sets SSK as the shared secret key
For SSE, sets SSK as the shared Secret key

Figure 1: Eun et al.'s privacy protection protocol

If verify the message m_3 successfully, B sets SSK as the shared secret key and computes $MacTag_B = f(SSK, ID_A, Q_A, Q_B)$ sends message $m_4 = \{MacTag_B\}$ to A. Otherwise, B stops the session.

5) Upon receiving the message m_4 and verify it successfully, A sets SSK as the shared secret key, if validation fails, A stops the session.

3.2 Debiao et al.'s Privacy Protection Protocol

In paper [7] Debiao et al.pointed out that Eun et al. protocol has the impersonation attack vulnerabilities. An attacker can impersonate even without knowing the private key of user A, so $Q_A^{'} = r_A G$, then $P_A = r_A Q_B^{'} = r_A r_B Q_B = r_A r_B d_B G$, $P_B = r_B d_B Q_A^{'} = r_B d_B r_A G$, $P_A = P_B$. Attacker and B could generate the same session key and $MacTag_A$ generated by attacker could pass B's verification. Besides, the TSM does not involve the above process. Then, the TSM cannot find the impersonation attack. Similarly, attacker can impersonate B. In order to overcome security weaknesses in Eun et al.'s protocol, Debiao et al. proposed the improvement privacy protocol, shown in Figure 2.

After receiving the user A's request for pseudonyms, the TSM generates n pseudonyms and sends them to A through a secure channel. The TSM also stores the user A's identity and pseudonyms into its database. Through the same method, the user A could get its pseudonyms and corresponding public/private keys.

User A and B require the pseudonymous set from the TSM and store them in the NFC device, randomly selected a pseudonym and the corresponding private key to start key agreement in the process of communication. The protocol can protect user's privacy and against impersonation attack, but NFC device requires additional costs for storage and computational efficiency decreased.

4 The Proposed Security Protocol

In view of the existing problems or shortcomings of above protocols, this paper proposes an improved privacy protection security protocol based on NFC, which uses Chebyshev-map and certificateless public key cryptography, TSM as a trusted third party to participate in the registration process and verify the true identity of the user when necessary. The protocol improves the computation efficiency, reduces the storage cost of NFC devices and can withstand impersonation attacks. This protocol includes the initialization phase and key agreement phase, and the process description is as follows.

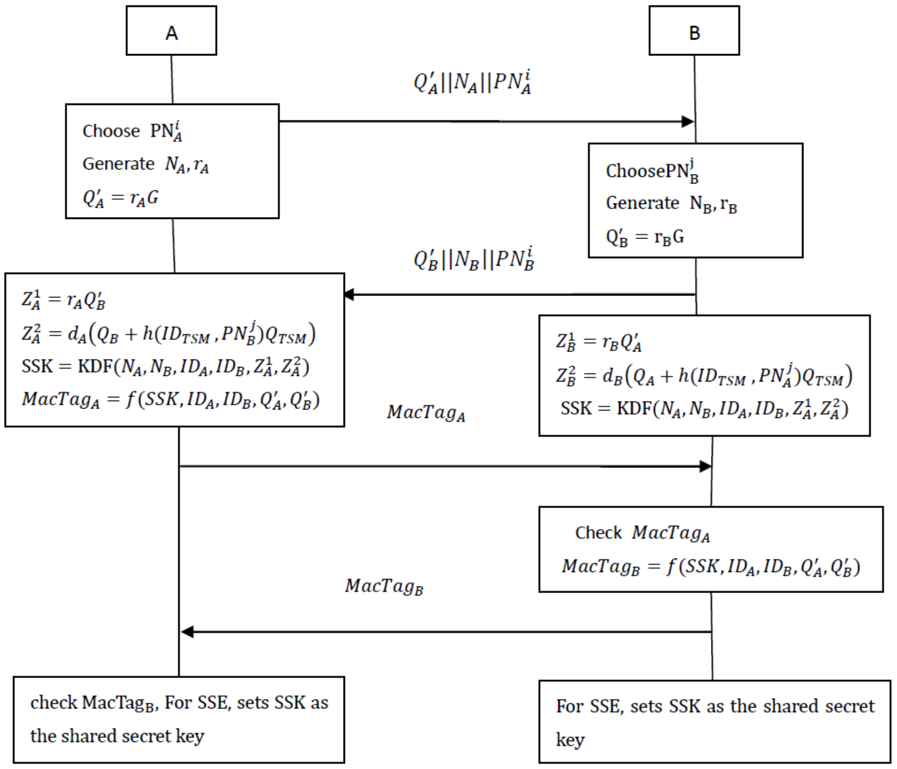

A
B
$Q'_A||N_A||PN_A^i$
Choose PN_A^i
Generate N_A, r_A
$Q'_A = r_A G$
ChoosePN_B^j
Generate N_B, r_B
$Q'_B = r_B G$
$Q'_B||N_B||PN_B^i$
$Z_A^1 = r_A Q'_B$
$Z_A^2 = d_A(Q_B + h(ID_{TSM}, PN_B^j)Q_{TSM})$
SSK = KDF($N_A, N_B, ID_A, ID_B, Z_A^1, Z_A^2$)
$MacTag_A = f(SSK, ID_A, ID_B, Q'_A, Q'_B)$
$Z_B^1 = r_B Q'_A$
$Z_B^2 = d_B(Q_A + h(ID_{TSM}, PN_A^j)Q_{TSM})$
SSK = KDF($N_A, N_B, ID_A, ID_B, Z_A^1, Z_A^2$)
$MacTag_A$
Check $MacTag_A$
$MacTag_B = f(SSK, ID_A, ID_B, Q'_A, Q'_B)$
$MacTag_B$
check $MacTag_B$, For SSE, sets SSK as the shared secret key
For SSE, sets SSK as the shared secret key

Figure 2: Debiao et al.'s privacy protection protocol

4.1 Initialization Phase

The premise of initialization phase is that the user A and B has been registered in the TSM. We assume that the user and the TSM mutual authentication has been conducted and the communication tunnel is secure. The process of register as follows:

1) A sends message $m_1 = \{q_A, ID_A\}$ to TSM ,q_A is the password generated by A and ID_A is A's identity.

2) Upon receiving the message m_1, TSM computes $R_A = T_{s_A}(x)$,s_A is the partial private key generated by certificateless public key cryptography [14].

In the key agreement and confirmation phase, when A sends a request to the TSM, TSM sends B's R_B to the A, the specific process is as follows:

1) A computes h($ID_A \| ID_B \| s_A$)and sends $m_1 = \{h(ID_A \parallel ID_B \parallel s_A) \parallel ID_A \parallel ID_B\}$ to TSM.

2) Upon receiving the request message m_1, TSM computes $h'(ID_A \parallel ID_B \parallel s_A)$ and if $h'(ID_A \parallel ID_B \parallel s_A)$ equal m_1, TSM computes $Enc(Q_A, R_B)$ and sends $m_2 = \{Enc(Q_A, Q_B) \parallel ID_A \parallel ID_B \parallel ID_{TSM} \parallel S_{TSM}\}$ to A. S_{TSM} is TSM signature on the message, $S_{TSM} = Sig(Enc(Q_A, Q_B) \parallel ID_A \parallel ID_B \parallel ID_{TSM})$.

3) A receives the message m_2 and verify the TSM signature S_{TSM} and decrypt $Enc(Q_A, R_B)$ by own private key d_A to get R_B.

Similarly, B gets R_A through these steps.

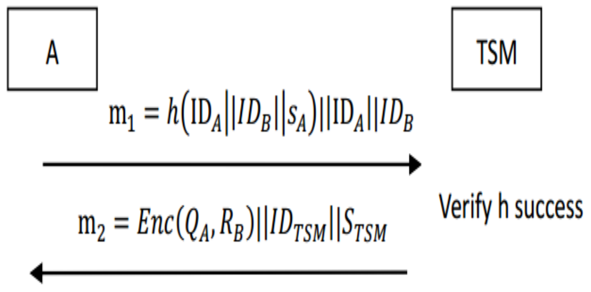

A
TSM
$m_1 = h(ID_A||ID_B||s_A)||ID_A||ID_B$
Verify h success
$m_2 = Enc(Q_A, R_B)||ID_{TSM}||S_{TSM}$

Figure 3: Initialization of the proposed protocol

4.2 Key Agreement and Confirmation Phase

Key agreement and confirmation process as shown in Figure 4, Specific steps are as follows:

1) A sends a request to TSM for getting R_B, generates a nonce N_A and a random number r_A, Then, A computes $Q'_A = r_A Q_A$, $Q''_A = r_A d_A Q_s + Q_A$ and sends the message $m_1 = \{Q'_A \parallel N_A \parallel Q''_A\}$ to B.

2) Upon receiving the message m_1, B sends a request to TSM for getting R_A, generates a nonce N_B and a

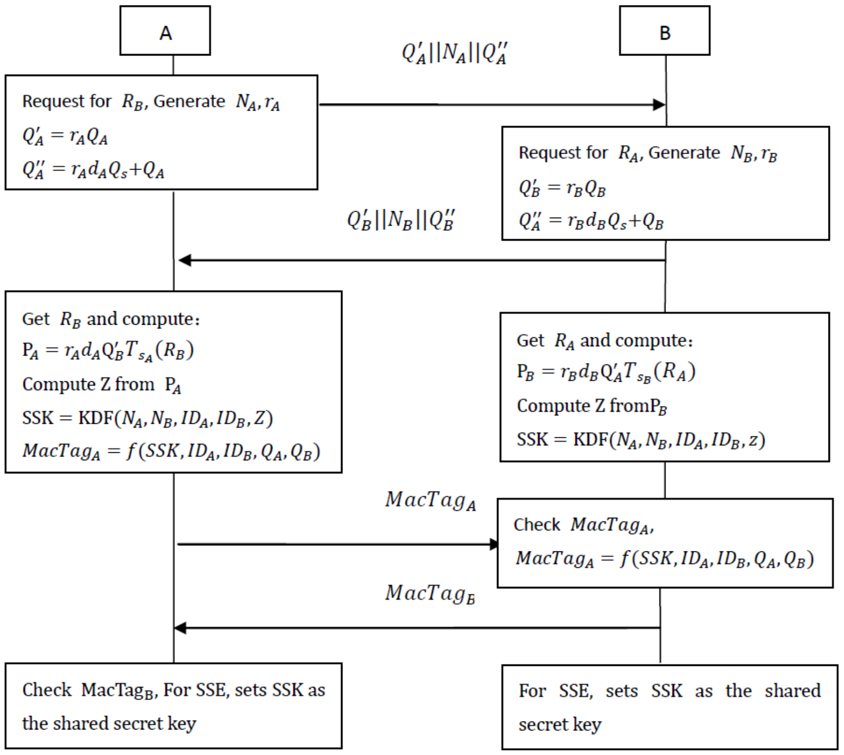

A
B
$Q'_A||N_A||Q''_A$
Request for R_B, Generate N_A, r_A
$Q'_A = r_A Q_A$
$Q''_A = r_A d_A Q_s + Q_A$
Request for R_A, Generate N_B, r_B
$Q'_B = r_B Q_B$
$Q''_A = r_B d_B Q_s + Q_B$
$Q'_B||N_B||Q''_B$
Get R_B and compute:
$P_A = r_A d_A Q'_B T_{s_A}(R_B)$
Compute Z from P_A
$SSK = KDF(N_A, N_B, ID_A, ID_B, Z)$
$MacTag_A = f(SSK, ID_A, ID_B, Q_A, Q_B)$
Get R_A and compute:
$P_B = r_B d_B Q'_A T_{s_B}(R_A)$
Compute Z fromP_B
$SSK = KDF(N_A, N_B, ID_A, ID_B, z)$
$MacTag_A$
Check $MacTag_A$,
$MacTag_A = f(SSK, ID_A, ID_B, Q_A, Q_B)$
$MacTag_B$
Check $MacTag_B$, For SSE, sets SSK as the shared secret key
For SSE, sets SSK as the shared secret key

Figure 4: The proposed key agreement and confirm protocol

random number r_B. Then, B computes $Q'_B = r_B Q_B$, $Q''_B = r_B d_B Q_s + Q_B$ and sends the message $m_2 = \{Q'_B \parallel N_B \parallel Q''_B\}$ to A.

3) Upon receiving the message m_2, A computes $P_A = r_A d_A Q'_B T_{s_A}(R_B)$, computes Z from P and $SSK = KDF(N_A, N_B, ID_A, ID_B, Z)$, $MacTag_A = f(SSK, ID_A, ID_B, Q_A, Q_B)$ and sends message $m_3 = \{MacTag_A\}$ to B.

4) Upon receiving the message m_3, B computes $P_B = r_A d_A Q'_B T_{s_B}(R_A)$, computes Z from P and $SSK = KDF(N_A, N_B, ID_A, ID_B, Z)$, If verify the message m_3 successfully, B sets SSK as the session key, computes $MacTag_B = f(SSK, ID_A, ID_B, Q_A, Q_B)$ and sends the message $m_4 = \{MacTag_B\}$ to A. Otherwise, B stops the session.

5) A receives the message m_4 and the validation is successful, sets SSK as the session, if validation fails, A stops the session.

5 Protocol Analysis

5.1 Security Analysis

Security analysis shows that the proposed protocol can provide anonymity and mutual authentication, be against to impersonation attacks etc. The specific analysis is as follows:

Security 1: The proposed security protocol could provide user anonymity.

Due to the wireless communication in NFC applications, the adversary C could control the communication channel totally. Then, he could intercepts the message $m_1 = \{Q'_A \parallel N_A \parallel Q''_A\}$, $m_2 = \{Q'_B \parallel N_B \parallel Q''_B\}$, $m_3 = \{MacTag_A\}$ and $m_4 = \{MacTag_B\}$ transmitted between the user A and the user B. But the identities of A and B are included in Q''_A and Q''_B separately. Without the A and B private key d_A and d_B, the adversary cannot get the identity of A or B. Thus, the proposed security protocol could provide user anonymity.

Security 2: The proposed security protocol could provide session key security.

Suppose the adversary C could get a session key generated in a previous session. He has to compute $P_A = r_A d_A Q'_B T_{s_A}(R_B) = r_A d_A r_B d_B G T_{s_A}(T_{s_B}(x)) = r_B d_B (r_A d_A G) T_{s_B}(T_{S_A}(x)) = P_B$ from $Q'_A = r_A Q_A$ and $Q'_B = r_B Q_B$. if he wants to get the session key in the current session since A and B generate new random numbers r_A and r_B for each session. Then, the adversary has to solve the computational DiffieHellman problem. Due to the hardness of the computational Diffie-Hellman problem, the proposed security protocol could provide session key security.

Security 3: The proposed security protocol could withstand impersonation attacks.

Table 2: Performance comparison

	The initiator user	The target user	Total
Chi et al.'s protocol	$2T_m + T_a + 2T_{AES} + T_{kdf} \approx 2606T_{Mul}$	$2T_m + T_a + 2T_{AES} + T_{kdf} \approx 2606T_{Mul}$	$4T_m + 2T_a + 4T_{AES} + 2T_{kdf} \approx 5212T_{Mul}$
Eun et al.'s protocol.	$3T_m + T_a + 2T_h + 2T_{Mul} + T_{kdf} \approx 3608T_{Mul}$	$3T_m + T_a + 2T_h + 2T_{Mul} + T_{kdf} \approx 3608T_{Mul}$	$6T_m + 2T_a + 4T_h + 4T_{Mul} + 2T_{kdf} \approx 7216T_{Mul}$
Debiao et al.'s protocol	$4T_m + T_a + 3T_h + T_{kdf} \approx 4806T_{Mul}$	$4T_m + T_a + 3T_h + T_{kdf} \approx 4806T_{Mul}$	$8T_m + 2T_a + 6T_h + 2T_{kdf} \approx 9612T_{Mul}$
Proposed protocol	$3T_m + T_a + 2T_h + 2T_{Mul} + T_{kdf} + T_{sym} \approx 3613T_{Mul}$	$3T_m + T_a + 2T_h + 2T_{Mul} + T_{kdf} + T_{sym} \approx 3613T_{Mul}$	$6T_m + 2T_a + 4T_h + 4T_{Mul} + 2T_{kdf} + 2T_{sym} \approx 7226T_{Mul}$

Let A and B be initiator user and target user separately. Assuming an attacker C fake A generates the message $m_1 = \{Q_A^{'} \parallel N_A \parallel Q_A^{''}\}$, N_A is a random number generated by C. After receiving the message m_1, B generate a nonce N_B and a random number r_B, compute $Q_B^{'} = r_B Q_B$ and send the message $m_2 = \{Q_B^{'} \parallel N_B \parallel Q_B^{''}\}$ to C. However, C does not calculate $P_A = r_A d_A Q_B^{'} T_{s_A}(R_B)$, because C does not have the partial private key s_A of A and private key d_A, so C can't generate the message $m_3 = \{MacTag_A\}$. B is able to find an attack by examining the validity of $MacTag_A$. Also attacker C cannot fake B. Thus, the proposed protocol could against impersonation attack.

Security 4: The proposed security protocol could withstand modification attacks.

Suppose the adversary C intercepts the message $m_1 = \{Q_A^{'} \parallel N_A \parallel Q_A^{''}\}$ and send it to user B after modification. Due to the difficulty of the computational Diffie-Hellman problem, C cannot generate the valid message $MacTag_A = f(SSK, ID_A, ID_B, Q_A, Q_B)$ where $P_A = r_A d_A Q_B^{'} T_{s_A}(R_B)$, Z computed from P_A, and $SSK = KDF(N_A, N_B, ID_A, ID_B, Z)$. Then, B could find the attack by checking the validity of $MacTag_A$. Similarly, it is clear that A could also find the modification attack. Thus, the proposed security protocol could withstand the modification attack.

Security 5: The proposed security protocol could withstand replay attacks.

Suppose the adversary C intercepts the message $m_1 = \{Q_A^{'} \parallel N_A \parallel Q_A^{''}\}$ and replay it to B, where $Q_A^{'} = r_A Q_A$. N_A and r_A are a nonce and a random number generated by A.We also suppose C replay the message $m_3 = \{MacTag_A\}$ to B upon receiving the message $m_2 = \{Q_B^{'} \parallel N_B \parallel Q_B^{''}\}$. However, B could find the attack by checking the validity of $MacTag_A$ since B generates a new nonce N_B for each session. From the similar steps, we could demonstrate that the user A could find the replay attack. Thus, the proposed security protocol could withstand the replay attack.

5.2 Performance Analysis

In this section, the computational cost and storage space of the proposed protocol is analyzed and also compared with Eun et al.'s protocol and Debiao et al.'s protocol in Table 2. Notations are defined as follows:

T_m: The running time of an elliptic curve point multiplication operation;

T_a: The running time of an elliptic curve point addition operation;

T_{Mul}: The running time of a modular multiplication operation;

T_h: The running time of a hash function operation;

T_{kdf}: The running time of a key derivation function operation;

T_{sym}: The running time of elliptic curve encryption or decryption.

According to the paper [1], the elliptic curve point multiplication operation is computational expensive and hash function operation is more efficient than other operations. To be specific, the following values are used:$T_m \approx 1200T_{Mul}$, $T_a \approx 5T_{Mul}$, $T_h \approx 0.36T_{Mul}$. Generally speaking, the key derivation function is constructed through hash function, so $T_{kdf} \approx T_h \approx 0.36T_{Mul}$. Elliptic curve decryption mainly involves addition operation, so $T_{sym} \approx T_a \approx 5T_{Mul}$. In [2], the protocol using AES encryption in the key agreement process. As we all know symmetric encryption algorithm is faster than asymmetric cryptography algorithm. So the running time of AES encryption algorithm is negligible compared to other computing time.

In the Debiao et al.'s protocol, NFC device need additional space to store pseudonym set. A pseudonym is composed of public key, private key(encrypted with long-term key of user), ID of TSM, and signature on the message. The size of the fields generally used in NFC protocol is shown in Table 3. The size of a signal pseudonym is computes as follows:

Size of PN=Public key + Encrypted Private Key + ID of TSM +Signature=1200its

Suppose the number of pseudonyms is 1000, the memory needed to store these much pseudonyms is 146.48Kbytes. However it is not a big deal, when considering the recent trends of combining mobile device and NFC. But updated environment is limited because of the billing charges of mobile devices.

Table 3: Size of the fields

Field	Size
N_A, N_B,r_A,r_B	96bits
Q_A,Q_B	200bits
$MacTag_A$,$MacTag_B$	96bits
SSK	128bits
d_A,Z	192bits
$Enc(Q_A, d_A)$	352bits
S_{STM}	448bits

According to the protocol analysis, Chi et al.'s protocol and Eun et al.'s protocol computation efficiency are the highest, but cannot against the impersonation attack. Debiao et al.'s protocol can against the impersonation attack vulnerabilities, but the computational efficiency is low and NFC devices need additional space to store pseudonym set. The proposed protocol can against the impersonation attack, improve the computational efficiency, reduce the storage space of the NFC device and computational efficiency is improved by 24.8% compared with Debiao et al.'s.

6 Conclusion

In this paper, we proposed an improved privacy protection security protocol, according to find out the deficiencies of the protocols from Eun et al. and Debiao et al. The analysis results of security and performance show that the improved protocol can provide user anonymity, session key security, resist against the modification attacks, withstand replay attacks and withstand impersonation attack. Furthermore, the improved protocol improves the computing efficiency and reduces the storage for NFC.

Acknowledgments

This work is supported by the science and technology project of Guangdong Province(No.2015B010128014, 2015B010108002, 2015B090906016, 2014A010103029 and 2014B090908011), and the Project of Guangzhou Science and Technology(No.201508010026).

References

[1] S. Chatterjee, A. K. Das, and J. K. Sing, "An enhanced access control scheme in wireless sensor networks," *Ad Hoc & Sensor Wireless Networks*, vol. 21, no. 2, pp. 121–149, 2014.

[2] Y. L. Chi, C. H. Chen, and I. C. Lin, "The secure transaction protocol in NFC card emulation mode," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 431–438, 2015.

[3] H. Eun, H. Lee, and H. Oh, "Conditional privacy preserving security protocol for NFC applications," *IEEE Transactions on Consumer Electronics*, vol. 59, no. 1, pp. 153–160, 2013.

[4] H. Eun, H. Lee, J. Son, and S. Kim, "Conditional privacy preserving security protocol for NFC applications," in *IEEE International Conference on Consumer Electronics (ICCE'12)*, pp. 380–381, Las Vegas, NV, Jan. 2012.

[5] Gartner, *Market Insight: The Outlook on Mobile Payment*, Technical Report, Market Analysis and Statistics, May 2010.

[6] P. Gope and T. L. Hwang, "Lightweight and energy-efficient mutual authentication and key agreement scheme with user anonymity for secure communication in global mobility networks," *IEEE Systems Journal*, vol. 99, no. 1, pp. 1–10, 2015.

[7] D. He, N. Kumar, and J. H. Lee, "Secure pseudonym-based near field communication protocol for the consumer internet of things," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 1, pp. 56–62, 2015.

[8] D. Huang, S. Misra, M. Verma, and G. Xue, "PACP: An efficient pseudonymous authentication-based conditional privacy protocol for VANETs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 7336–746, 2011.

[9] ISO/IEC, *Information Technology – Telecommunications and Information Exchange Between Systems – NFC Security – Part 1: NFC-SEC NFCIP-1 Security Services and Protocol*, ISO/IEC 13157-1:2010.

[10] ISO/IEC, *Information Technology – Telecommunications and Information Exchange Between Systems – NFC Security – Part 2: NFC-SEC Cryptography Standard Using ECDH and AES*, ISO/IEC 13157-2: 2010.

[11] Juniper Research, *NFC Mobile Payments and Retail Marketing: Business Models & Forecasts 2012-2017*, Tenical Report, May 2012.

[12] J. H. Lee, J. Chen, and T. Ernst, "Securing mobile network prefix provisioning for NEMO based vehicular networks," *Mathematical and Computer Modelling*, vol. 55, no. 1-2, pp. 170–187, 2012.

[13] J. Li, S. Li, and Z. Chen, "Cryptanalysis and improvement of lai's authenticated group key transfer protocol," *Application Research of Computers*, vol. 32, no. 1, pp. 254–257, 2015.

[14] X. Niu, S. Guo, and Y. Wang, "Elliptic curve lightweight authentication and key agreement scheme," *Computer Science*, vol. 42, no. 1, pp. 137–141, 2015.

[15] Q. Qian, Y. L. Jia, and R. Zhang, "A lightweight RFID security protocol based on elliptic curve crytograph," *International Journal of Network Security*, vol. 18, no. 2, pp. 354–361, 2016.

[16] S. Wang, S. Liu, and D. Chen, "Security analysis and improvement on two RFID authentication protocols," *Wireless Personal Communications*, vol. 82, no. 1, pp. 21–33, 2015.

[17] X. Wang, Z. Huang, Q. Wen, and H. Zhang, "An efficient anonymous batch authenticated and key agreement scheme using self-certified public keys in VANETs," in *IEEE Region 10 Conference on TENCON*, pp. 1–4, Xian, Oct. 2013.

[18] C. H. Wei, M. S. Hwang, and A. Y. H Chin, "A mutual authentication protocol for RFID," *IEEE IT Professional*, vol. 13, no. 2, pp. 20–24, 2011.

[19] C. H. Wei, M. S. Hwang, and Y. H. Chin, "An improved authentication protocol for mobile agent device in RFID," *International Journal of Network Security*, vol. 10, no. 5, pp. 508–520, 2012.

[20] Z. Wu, "Research on privacy-preserving key technologies of the NFC mobile application system," *The PLA Information Engineering University*, 2012.

Jie Ling received his Ph.D degree in computation mathematics from Sun Yat-sen University (China) in June 1998. He is a professor in computer science in Guangdong University of Technology. His current research interest fields include computer applications and Intelligent video processing technology.

Ying Wang received her bachelor's degree from University of Jinan in China in June 2013. She is currently a master degree candidate in Guangdong University of Technology (China). Her current research interest includes network security protocols, cloud computing security and big data security.

Weifeng Chen received Ph.D. degree in Computer Science from University of Massachusetts at Amherst in September 2006. He is also a tenured associate professor in Department of Mathematics and Computer Science in California University of Pennsylvania. His current research interest includes cyber security, privacy protection, artificial intelligence and big data security.

Mojette (d) Secret Image "SEDIH" in an Encrypted Double Image - A Histo Approach

Padmapriya Praveenkumar, Karuppuswamy Thenmozhi, John Bosco Balaguru Rayappan and Rengarajan Amirtharajan

(Corresponding author: Rengarajan Amirtharajan)

School of Electrical & Electronics Engineering, SASTRA University

Thanjavur-613 401, India

(Email: amir@ece.sastra.edu)

(Received Sept. 29, 2015; revised and accepted Dec. 7, 2015 & Jan. 15, 2016)

Abstract

In this paper, double image encryption technique has been considered to carry secret data using bit plane concept. The Mojette Transformed Huffman Encoded (MOTHE) secret logo image was hidden in the difference image generated by histogram approach formed from the double images. The MOTHE secret logo enhances authentication, security and compression of bits as compared with the traditional embedding algorithms. In this proposed security scheme, transformed secret sequences and reversible data hiding in an encrypted double image were employed to enhance the sternness of the security barrier. The number of bits embedded and it's PSNR (Peak Signal to Noise Ratio) for various images in 512 × 512 and 256 × 256 have been estimated using MATLAB and compared with the available literature.

Keywords: Bit-plane image encryption, histogram modification, Huffman encoding, Mojette transform, reversible data hiding

1 Introduction

Unlike traditional methods, digital era communication habitually consents one to determine one's own level of security. As per one of the many, the means of ensuring that the data is solely available to those who are at liberty to use it, that data can only be altered by those who are meant to do so with the ease of nominal and managerial measures is termed as information security will further classified as cryptography, watermarking, image encryption and steganography. Security attack is a doable concern which is addressed over and over, but even with unswerving endeavors there exist ambiguities in the security system. Of all, crackers and intruders pose a prime threat to secret communication. The information transmitted wireless is exposed to a lot of threats and hence security measures are mandatory.

Reversible data hiding scheme unlike steganography provides reconstruction of the cover and embedded secret data also. Reversible data hiding was introduced and implemented by Ni et al. in 2006 [10] to embed secret data in difference image by improving the PSNR and embedding capacity and they proposed a data hiding technique that's completely reversible and was based on the modification of the histogram. Kuo-Liang Chung et al. [1] introduced a watermark block based complement scheme to decrease the distortion in reversible data hiding. Steganalysis is done to identify the cover and embedded data. Der Chyuan et al. [2] proposed a Steganalysis scheme to identify the hidden data and cover image based on histogram feature coding approach that detects the presence of steganographic data. In recent years the authentication and security of the transformed encrypted images are very efficient when Mojette transform was utilized as it concentrates more on cryptography, watermarking and in compression [2, 3].

The innate relationship between the adjacent pixel information is used to attain the difference between pixels using sub-sampled images to form the resultant histogram are concentrated by Kyung-Su Kim et al. in 2009 [5]. Mohankumar and Shanmuganathan [8] proposes a data embedding technique utilizing high capacity and provides several security levels. Here, the cover image is used to hide and then the stego file is obscured in one more image. Survey on data hiding techniques and principles that uses reversible concept was presented by Masoud Nosrati et al. [12] in which data hiding techniques like Pair-Wise Logical Computation (PWLC) and data hiding by Template ranking with symmetrical Central pixels (DHTC) technique has been considered. Dual image encryption making use of Hill cipher to promote entropy was carried out by Panduranga et al. [14].

Reversible data hiding for encrypted images was carried out to improve security and authentication. An algorithm which includes compression, encryption and embedding and make use of reversible data hiding concept and improves the PSNR value [15, 16]. Rajendra, Kanphade and Narawade [17] proposed a Forward Modified Histogram Shifting (FMHS) that has reduced the shifting of the pixels and yields high embedding capacity. Most of the factors with PSNR and embedding capacity included have been optimized in this proposed method. In 2012, Ramaswamy and Arumugam [18] projected a Data Hiding scheme that based on the shifting of histogram and completely lossless which accounts for over and under flow problems in pixel values and climbed that colour image embedding provides more embedding capacity as compared to gray scale images.

Raju, David and Rao [19] proposed an algorithm implementing histogram peak and zero points. It is grounded on the binary tree approach for multiple of peak points with histogram shifting for every overflow and underflow. In 2009, data hiding algorithm based on histogram approach utilizing difference of pixels, difference expansion and histogram shifting technique was proposed by Tai et al. [24]. Here, the distribution of pixel differences results not only on large embedding capacity but gives very low deformation. Further, this algorithm also prevents overflow and underflow problem.

Data hiding scheme that uses multi-dimensional and multi-level shifting of histogram has been proposed by Wang et al. [25]. Xinlu and Yang proposes a high capacity and adaptive embedding data hiding based on prediction-error has been considered [4]. A multilevel histogram modification scheme for embedding secret data was proposed by Zhao et al. [26] that modifies the histogram constructed based on the neighbour pixel differences instead of the host images histogram.

Mojette is the well known word of the city of Poitiers, France which means white beans. It was adopted as a standard tool for addition and subtraction computations by the children living there. Initial work on MOT was developed by IRCCyN laboratory, France in 1994. In 1995 the first work on MOT was published. The main aim of MOT is to determine the projections on the image. Radon transform serves as an application of discrete geometry and one best application and replica of it is MOT. It is characterized as specified for rational projection angles [20, 22]. Co generic to radon transform, MOT is also used to embody an image as a set of projections and every finite discrete projection has got its own inverse. It adopts major properties of radon transform and apart from those it has got a noticeable property of redundancy [7].

MOT projections can usefully be modified for the application of compressed sensing. This combination mainly has an outbreak in the reconstruction frame. Hence MOT along with compressed sensing produces effective outcomes of reduced radiation dosages without affecting the image quality. Watermarking the image on the whole only the projections are watermarked [6, 11].

David A. Huffman, in the year 1952, from MIT introduced the finest Huffman entropy coding. The main of his invention is to construct a code to provide minimum redundancy and to provide lossless compression of data. Huffman based text steganography was carried out by Satir and Isik, [21] to provide an increase in compression ratio besides the security features. Steganography methods can be classified as spatial domain [9] or transform domain [23] and few resist statistical steganalysis [13].

But majority of these schemes make use of histogram approach with either gray scale or colour images to improve the hiding capacity and PSNR. In this work, two grey scale images were formed by using bit plane concept using double image encryption. Here the histogram of the second image has been constructed based on the pixel difference from the first image. Then in the resultant image, MOTH secret logo has been embedded in the peak point to improve the embedding capacity and PSNR of the proposed scheme. This improves compression of secret data as compared with the traditional schemes. The PSNR and the number of bits embedded of the proposed algorithm were compared with the available literature and found to be better. In total, this study highlights the following;

1) Reversible data hiding;
2) Double image technique involving bit plane concept;
3) The Mojette Transformed Huffman Encoded (MOTHE) secret logo were hidden;
4) It provides high PSNR, embedding capacity and compression of bits as compared with the available literature;
5) Histogram approach formed from double images.

2 Preliminaries

2.1 Reversible Data Hiding

It is a technique, where secret data bits were embedded into a cover as traditional secret communication and includes the extraction of the secret data and the cover medium. The performance metrics can be analyzed using the complexity of the key involved, visual quality and the payload capacity.

2.2 Histogram Shifting

This method embeds the secret data in the cover media considering and analyzing the histogram of the image by shifting process.

2.3 Peak Point Embedding

This method finds either the peak or zero points in the histogram. Secret data hiding is carried out by shifting these peak or zero points resulting in maximum payload capacity with minimal distortion. It avoids overflow (pixel value going beyond 255) and underflow (Pixel value below 0) problems.

3 Proposed Methodology

In the proposed scheme, double image encryption has been concentrated, in which two 512 × 512 grey scale images were used to form a single image using bit plane concept. Then based on the pixel difference from the first image, the histogram of the second image was formed. To the resulting image, MOTH secret logo has been embedded in the peak point to improve the embedding capacity and PSNR. The proposed reversible data hiding encryption and decryption schemes are given in Figures 1 and 2 respectively.

3.1 Encryption Algorithm

1) Get the input cover images C1 & C2.
2) Separate C1 & C2 into its corresponding bit planes.
3) Replace the LSB bit planes of C2 with LSB bit planes of C1 and store it as A1.
4) Replace the LSB bit planes of C1 with LSB bit planes of C2.
5) Read the image matrix A1 and A2.
6) Resize A1 & A2 to 512 × 512 (P × Q), where P, Q represents the size of A1 & A2.
7) Convert A1 & A2 to double values.
8) Compute histogram to image matrix A1 & A2.
9) Divide the image into 4 × 4 blocks.
10) Compute the difference between adjacent columns.

$$\begin{aligned} A1(i,j) &= |A(i,j) - A(i,j+1)|; \\ &\quad 0 \leqslant i \leqslant P, 0 \leqslant j \leqslant Q-2. \\ A2(i,j) &= |B(i,j) - B(i,j+1)|; \\ &\quad 0 \leqslant i \leqslant P, 0 \leqslant j \leqslant Q-2, \end{aligned}$$

where A1 & A2 are the difference block of size $P \times Q - 1 (4 \times 3)$. A, B are the 4 × 4 image block of A1 & A2. i, j are the rows and columns of Z.

11) Compute the difference between resultant image A1 & A2.

$$Z(i,j) = A1 - A2;$$

12) Compute the histogram of Z (i, j).
13) Record the peak point p in the histogram of Z (i,j), where V represents the peak point that has larger no of pixel values.
14) If Z(i, j) greater than peak point then Z'(i, j)= Z(i, j)+1.

$$Z'(i,j) = \begin{Bmatrix} Z(i,j)+1) & if\ Z(i,j) > V \\ Z(i,j) & otherwise \end{Bmatrix}$$

For $0 \leqslant i \leqslant P$, $0 \leqslant j \leqslant Q-2$, where Z'(i, j) = difference image.

15) The principle Z'(i, j) is applied for each image blocks.
16) If the pixel value is equal to V can be modified to hide U, where U represents the secret data bits.
17) Then the condition for hiding message in each block is given by,

$$Z''(i,j) = \begin{Bmatrix} Z'(i,j)+U) & if\ Z'(i,j) = V \\ Z'(i,j) & otherwise \end{Bmatrix}$$

For 0 ⩽ i⩽ P, 0⩽ j⩽ Q-2, where Z"(i, j) is the modified hidden difference image, U represents the secret data bits.

18) Performing inverse transformation J^{-1} to each 4×3 block of difference image and construct marked image.

$$W(i,1) = \begin{Bmatrix} A(i,2) + Z''(i,2) & if\ A(i,1) > A(i,2) \\ A(i,1) & otherwise \end{Bmatrix}$$

For 0 ⩽ i⩽ P, 0⩽ j⩽ Q-2.

$$W(i,2) = \begin{Bmatrix} A(i,1) + Z''(i,1) & if\ A(i,1) \leqslant A(i,2) \\ A(i,2) & otherwise \end{Bmatrix}$$

For 0 ⩽ i⩽ P, 0⩽ j⩽ Q-2.

3.2 Mojette Transform (MOT)

The main property of MOT is only additions, subtractions and apparently the initial discrete information can be distributed into multiple projections. Reconstruction of initial information can also be performed when ample projections are available. The main objective is to derive certain inconsistent set of information from projections to avoid rotation attacks. This serves to be a useful cryptographic scheme. The linearity property of the transform is much helpful in decoding the cryptic image where it takes modulus addition of pixels.

The MOT projects the original image block as,

$$A = A(b,c); \quad b = 1...D; \quad c = 1...E$$

On a set of projections

$$E = \{E_g(h) \quad g = 1, ...g \quad h = h...h_g\}$$

It's a version DRT for a group

$$E_g(h) = projection \quad (J_f, K_f, l_f)$$

where J_f and KJ_f are the projection lines.

$$E_g(h) = \sum^{b,c \sum H} A(b,c) \quad \partial(b_l - iq_k - jp_k)$$

where,

$$\partial(a) = \begin{Bmatrix} 1 & if \quad a = 0 \\ 0 & otherwise \end{Bmatrix}$$

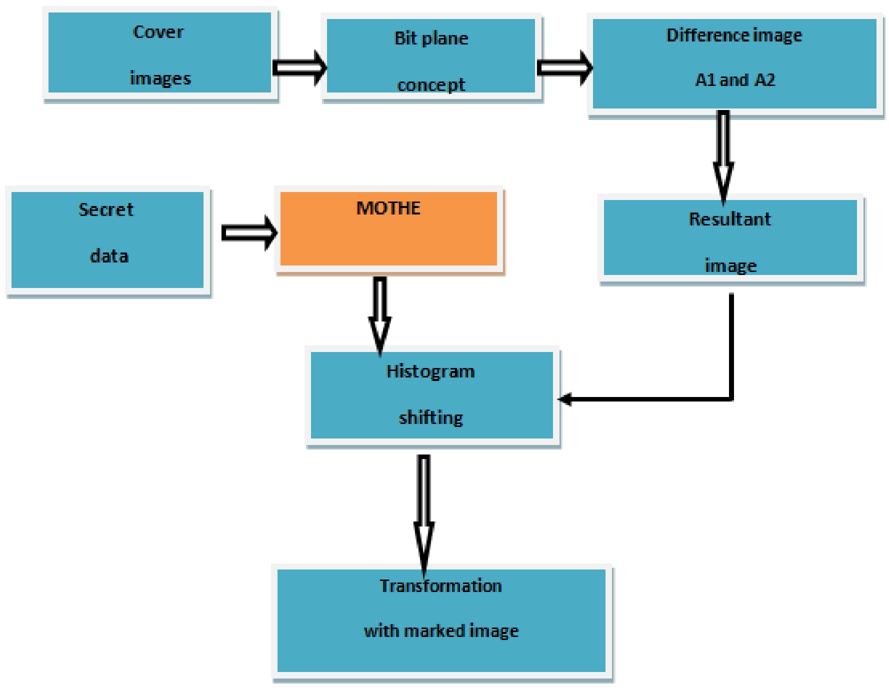


Figure 1: Encryption scheme

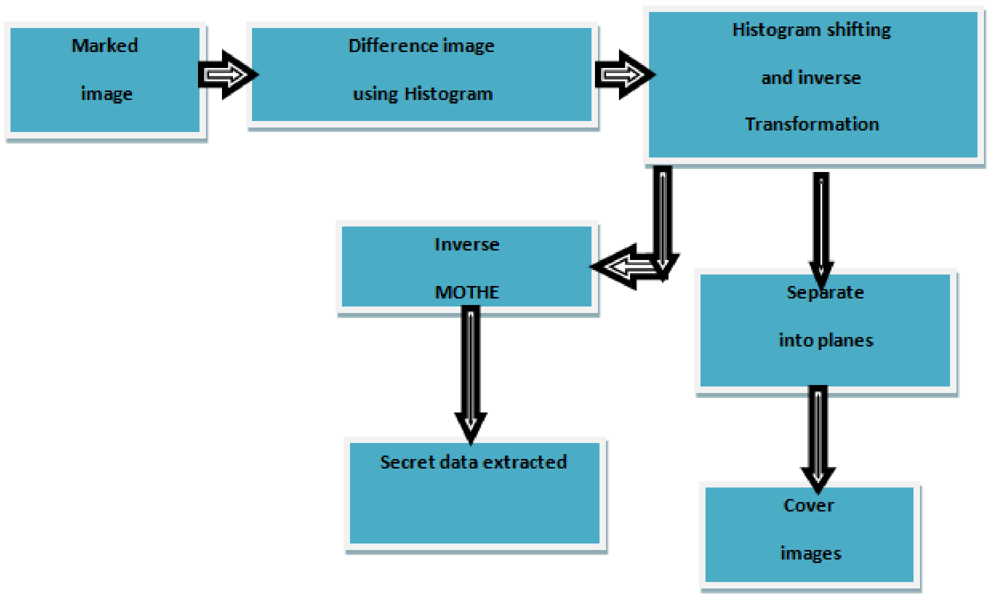


Figure 2: Decryption scheme

$$H = (b, c;\quad l_h - bk_f - cj_f = 0)$$

H represents the digital bin in ϕ_f direction and on l_h Then the total number of bins will be calculated by

$$y_b = (D-1)\,|J_f| + (E-1)\,|K_f| + 1$$

The projection angles are given by $\theta_i = \tan^{-1}(j_a/i_a)$, where the set of vectors (j_a/i_a) should be co-prime and ia should always be positive except for a single case of (1,0). The transformed image will have a set of projections of which every element is termed as bins. These bins are obtained by addition of pixels along the line of projection. In order to recover the image an iterative process of search and update of one-one pixel-bin is performed. Back-projection of this bin value onto the pixel and consequent subtraction in other respective projections will be done. Then the pixel values from the projection bins are chosen and the unwanted bins are discarded during reconstruction. This reduces redundancy of the projection bins.

3.3 Secret Logo Embedding

1) Read the image matrix S.
2) LSB bit-planes of S are combined as N (512 × 512 image).
3) Difference between the adjacent pixel values was taken.
4) Read the difference image as M.
5) M is subjected to Mojette Transform, followed by Huffman encoding.
6) As an example, consider a 3×3 matrix from the secret data to be embedded as in Table 1.

Table 1: 3 × 3 Secret data

4	6	1
2	2	0
3	5	8

7) Calculate the bin values as in Figure 3.
8) For retrieval of the original data at the receiver end, bins 1, 5, 6,7, 8, 9, 10, 11 and 13 are required.
9) Among 13 bin values estimated, only 9 bins are used neglecting the 4 bin values.
10) For the 9 bin values (3, 1, 4, 8, 6, 5, 8, 9, 9), calculate the mean value and the mean value is found to be 6.
11) Then from the calculated mean value, find the difference between the mean and the bin values, calculated value= mean value − bin value.
12) The calculated values are -3, -5, -2, 2, 0, -1, 2, 3, 3.

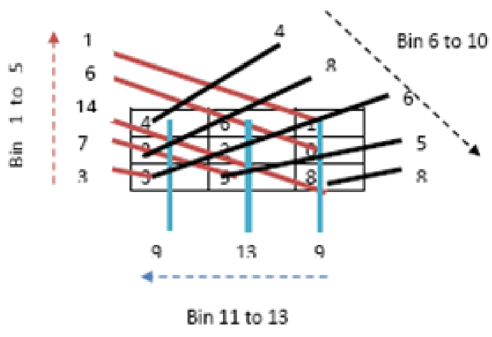


Figure 3: Bin values calculation

Table 2: Bin values and their estimated probability values

Calculated bin values	Probability
-3	0.5
-5	0.05
-2	0.05
2	0.125
0	0.05
-1	0.05
2	0.125
3	0.25
3	0.25

13) To provide compression, Huffman encoding was applied to the calculated values.
14) Distribute the probability for the calculated values as in Table 2.
15) The Calculated bin values are encoded using Huffman encoding procedure as shown in Figure 4.
16) The encoded bits are 1110000100001101111111111000 11001.
17) Totally 32 bits are encoded.
18) For a 3 × 3 matrix, normally 9 × 8=72 bits will be transmitted; instead it has been reduced to 32 bits by applying MOHTE.

3.4 Decoding of the Secret Data Bits

1) From the transmitted sequence, the calculated values are obtained using Huffman tree.
2) The calculated values are -3, -5, -2, 2, 0, -1, 2, 3, 3.
3) Then to the calculated value mean will be added to determine the original bin values. Calculated value + Mean value = Original bin value.
4) Then the original bin values are recovered as 3, 1, 4, 8, 6, 5, 8, 9, 9.

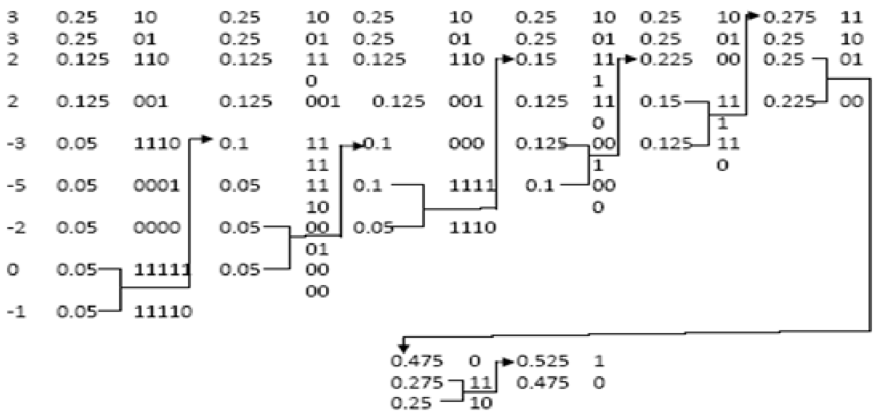


Figure 4: Bin value calculation using Huffman encoding procedure

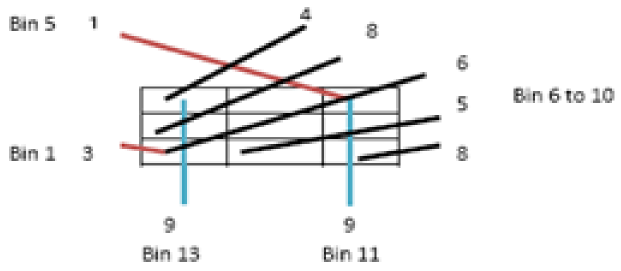


Figure 5: Arrangement of bin values

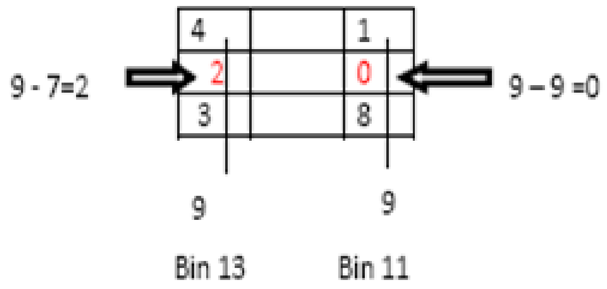


Figure 6: Bin values estimation

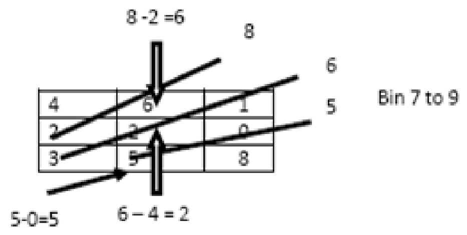


Figure 7: Secret matrix formation

5) Then arrange the original bin values in respective positions to extract the matrix values as shown in Figure 5.

6) Using the bin values 1, 5, 6 and 10 the corner value of the matrix was obtained.

7) Then using the bin values 13, 11 and using Step 19, next two matrix elements are calculated as shown in Figure 6.

8) By using the bin values, 7, 8 and 9 the remaining matrix elements was obtained as shown in Figure 7.

9) Thus the original secret 3 *times* 3 matrix was reconstructed.

3.5 Decryption

Perform the inverse operation of encryption to extract the cover images 1 and 2 and the secret data bits.

3.6 Secret Logo Image to be Embedded

Secret logo image to be embedded is given in Figure 8. From Table 4 it is clear that, for a 256 $\times$ 256, the original data to be embedded will be 65, 536 bits and after applying Mojette Transform (MOT) it will be reduced to 10, 752 and still reduced to 3392 bits after applying

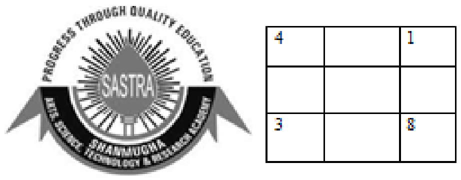

PROGRESS THROUGH QUALITY EDUCATION
SASTRA
SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
4
1
3
8

Figure 8: Secret logo to be embedded after applying MOTH

Mojette Transform Huffman encoding (MOTH). Then for 512 × 512 image, the original data to be embedded will be 262144 bits and after applying Mojette Transform (MOT) it will be reduced to 21, 504 and still reduced to 6784 bits after applying MOTH.

Table 3: Comparison of Mojette transform with traditional embedding

Secret logo	Bits to be embedded		
	Without MOHTE	With MOT	With MOTHE
256× 256	65,536	10,752	3392
512×512	2,62,144	21,504	6784

4 Results and Discussion

The proposed algorithm was based on the bit plane based reversible data hiding on double images. Figure 8 shows the secret logo to be embedded after applying MOTH. Here two cover images of camera man and moon are taken as input as shown in Figure 9a and Figure 9b. Difference images of 9(a) and 9(b) are given in Figure 10a and b respectively. Then the difference image between 10a and b was computed and the resultant image and its histogram were shown in Figure 11a & Figure 11b respectively. Figure 12a represents the secret data bits embedded in 11(a) and its corresponding histogram in Figure 12b.The marked image was given in Figure 13a and its histogram in Figure 13b using inverse transformation and then Gaussian noise with zero mean and 0.02 variance is added to the marked image and was given in Figure 14a and its histogram in Figure 14b.The decrypted image was found to be robust even after adding noise.

Figure 15a provides the decrypted image after adding Gaussian noise and its histogram in Figure 15b.

From Figure 15a it is revealed that even after adding noise decryption is possible. Figure 16a, b, c, d and e provides the difference image, marked image, marked image with Gaussian noise, final decrypted cover image 1 and the final decrypted cover image 2 respectively. Figure 17 a, b, c, d and e provides the Lena image formed from bit-planes of double image, its difference image, marked image, marked image with Gaussian noise, final decrypted

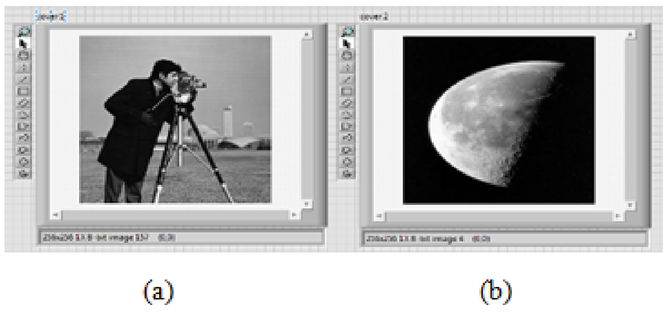


Figure 9: a) Cover image C1; b) Cover image C2

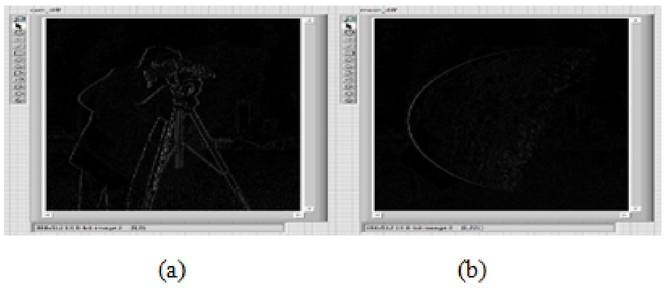


Figure 10: a) Difference image of 9(a); b) Difference image of 9(b)

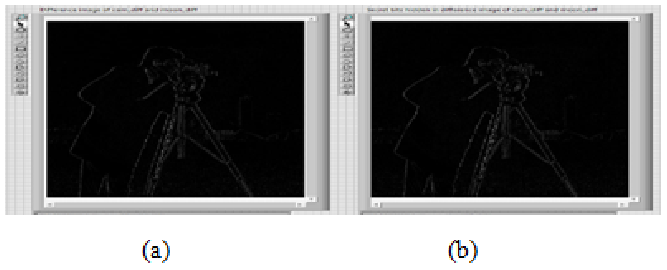


Figure 11: a) Difference image of 10(a & b); b) Secret data bits hidden in 11(a)

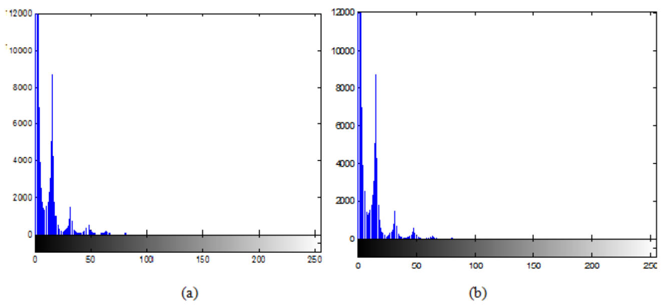


Figure 12: a) Histogram of 11(a); b) Histogram of 11(b)

Table 4: Performance metrics for various images of size 512 × 512

Images (512 × 512)	Number of pixels embedded	PSNR
Boat	49297	48,816
Barbara	36919	48.8380
Jet	51233	48.7596
Pout	57514	49.5636
Lena	50580	48.8000
Cell	60141	49.7397
Cameraman	51801	48.9513
Baboon	38091	48.4060
Average PSNR		49.06 dB

Table 5: Comparative analysis of PSNR in dB

Host image	Xinlu, Li and Yang.(2013)	Ratna Raju et al., (2010)	Ramaswamy et al., (2012)	Proposed
Lena	42.71	44.28	37.79	48.8
Jet	46.03	44.02	36.24	48.75
Boat	37.81	44.01	39.67	48.8
Baboon	2.01	44.19	36.24	48.4

Table 6: Comparison of payload characteristics I

Host image	Payload (bpp) (Ni et al., 2006)	Proposed	Increased payload (in percentage)
Lena	5460	50580	826
Boat	7301	51233	601
Baboon	5421	38091	603
Average	6060	46634	669

Table 7: Comparison of payload characteristics I

Host image	Payload (bpp) (Ratna Raju, David and Prasada Rao, 2010.)	Proposed	Increased payload (in %)
Lena	22390	50580	125.90
Baboon	25530	38091	49.200
Average	23960	44335	85.03

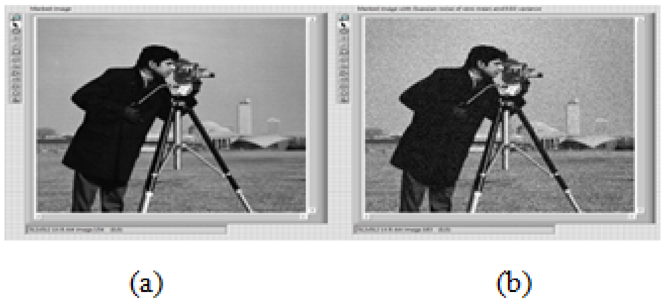


Figure 13: a) Marked image; b) Marked image with Gaussian noise of zero mean and 0.02 variance

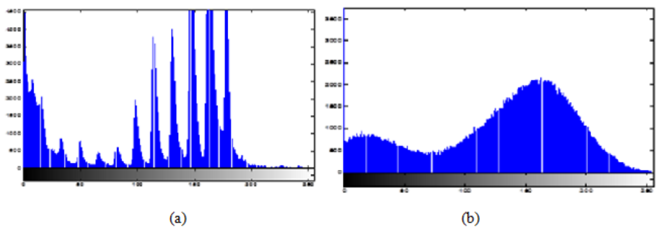


Figure 14: a) Histogram of 13(a); b) Histogram of 13(b)

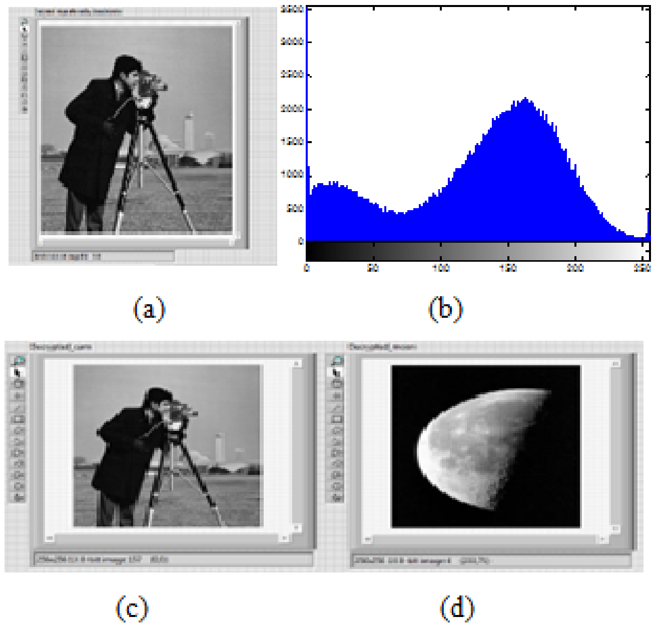


Figure 15: a) Decrypted cover image 1 affected by Gaussian noise; b) Histogram of 15(a); c), d) Decrypted cover images 1 and 2

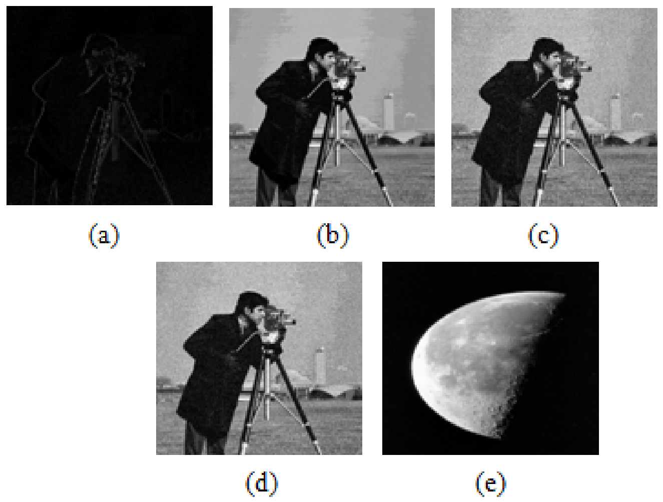


Figure 16: a) Difference image; b) Marked image; c) Marked image with Gaussian noise; d) Decrypted cover image 1; e) Decrypted cover image

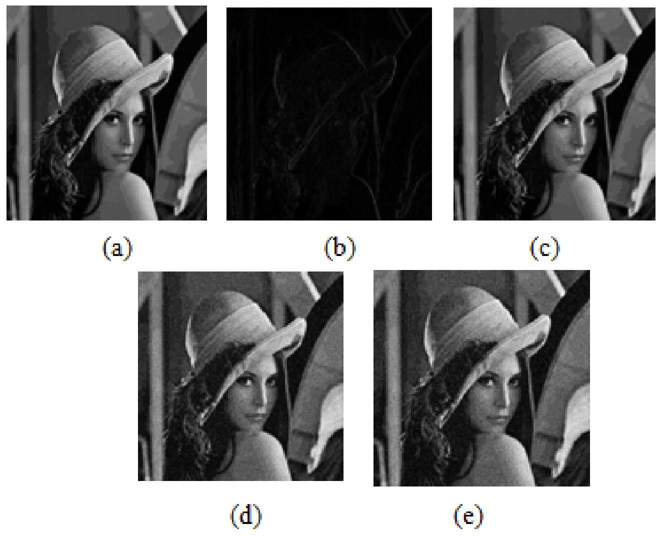


Figure 17: a) Original biplane hidden image; b) Secret bits embedded in 17 (a); c) Marked image; d) Marked image with Gaussian noise; e) Decrypted cover image 1

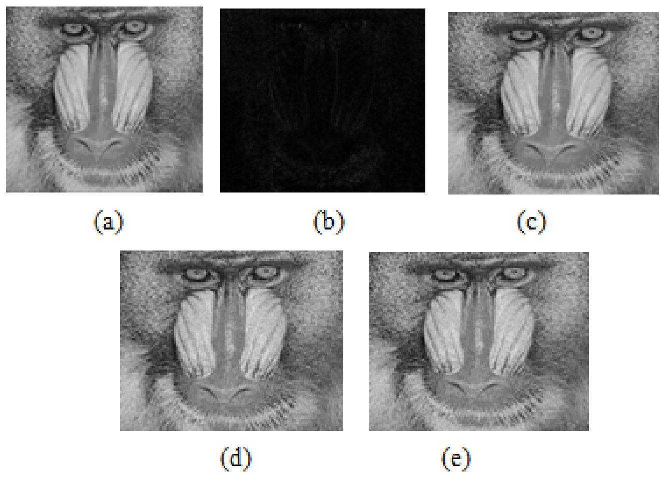


Figure 18: a) Original biplane hidden image; b) Secret bits embedded in 18(a); c) Marked image; d) Marked image with Gaussian noise; e) Decrypted cover image 1

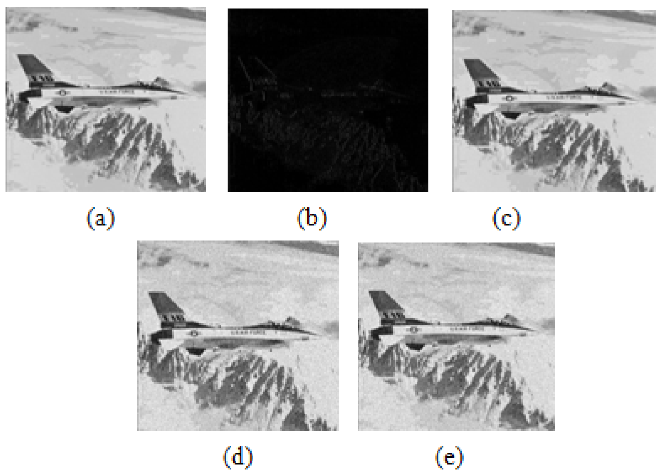


Figure 19: a) Original biplane hidden image; b) Secret bits embedded in 19(a); c) Marked image; d) Marked image with Gaussian noise; e) Decrypted cover image 1

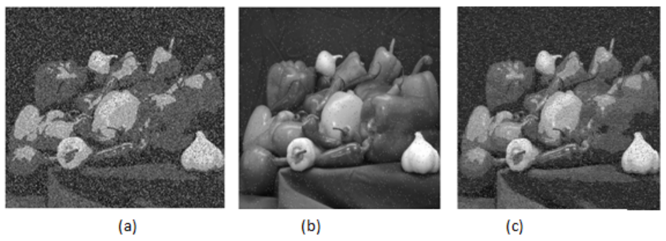


Figure 20: Decrypted image after a) Gaussian filtering attack; b) pepper-salt noise attack; c) median filtering

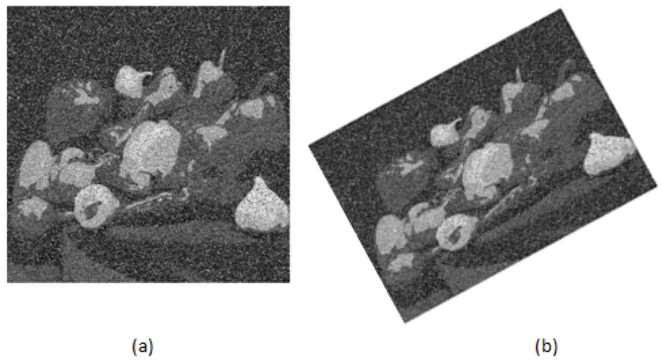


Figure 21: Decrypted image after a) cropping and b) rotation

cover image 1 respectively.

Figure 18 a, b, c, d and e provides the Baboon image formed from bit-planes of double image, its difference image, marked image, marked image with Gaussian noise, final decrypted cover image 1 respectively. Figure 19 a, b, c, d and e provides the plain image formed from bit-planes of double image, its difference image, marked image, marked image with Gaussian noise, final decrypted cover image 1 respectively. For all the test images considered, moon image was considered to be the second cover image. Figure 20 a, b and c represents the decrypted image after applying gaussian, pepper-Salt and median noise attacks respectively. Figure 21 a and b represents decrypted image after cropping and rotation attacks respectively Table 5 provides the information about the number of bits embedded and its PSNR for various images in 512 × 512 format. Table 6 estimates the PSNR of the proposed scheme with the available literature and found to be better. Table 7 and 8 provides the comparison of payload characteristics considering various images like Lena, Boat and Baboon in comparison with (Ni et al., 2006 and Ratna Raju, David and Prasada Rao. 2010) respectively.

5 Performance Analysis

5.1 PSNR & MSE

PSNR is a measure to validate the image quality after embedding. It is given by

$$\begin{aligned} PSNR &= 10 \times log_{10} \frac{I^2}{MSE} \\ MSE &= \frac{1}{MN} \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} [I()i,j) - K(i,j))]^2 \end{aligned}$$

where M and N represents the row and column of the image matrix and I take the value 255.The average PSNR value of proposed method is found to be 49.06dB which is superior when compared to (Ramaswamy and Arumugam. 2012; David and Prasada Rao. 2010 and Xinlu, Li and Yang. 2013).

5.2 Embedding Capacity

Embedding capacity provides the hiding capacity of the algorithm. The embedding capacity depends on the pixel values at the peak point Embedding capacity = number of pixels at peak point of difference image. The estimated values of the proposed method are found to be better with the previous work in payload (Ni et al., 2006 and Ratna Raju, David and Prasada Rao. 2010).

6 Compression Achieved through MOHT

The secret logo to embed was passed through MOHTE. For a 4×4 image, the number of bins will be 21 by applying MOT. Then each bit will be represented by 8 bits, so totally 21 × 8=168 bits are required for embedding. But out of the 21 bins, 16 bins are sufficient to retrieve the original 4×4 image. So it will be reduced to 16 × 8=128 bits results in the removal of 40 redundant bits.

Then to this 16 bin values, Huffman encoding has been applied to remove redundant bits. The resultant bits after MOTH will be 53 bits. Thus an increase in the compression of 31 percentage will be provided.

So, for a 256 × 256 image, there will be hardly 64 numbers of 4×4 blocks will be there. So the required bits are 64 × 53 bits = 3392 bits. Similarly for a 512 × 512 image, there will be hardly 128 numbers of 4 × 4 blocks will be there. So the required bits are 128 × 53 bits = 6784 bits.

7 Conclusions

In this paper, bit planes of the double images have been formed, and then difference image was computed based on histogram approach from the double images. Then MOHT secret logo image was hidden in the difference image to improve the hiding capacity in spatial domain. This embedding provides compression, authentication and security as compared with the traditional embedding schemes. The proposed methodology employs grey scale images and implemented using MATLAB. The results demonstrate that embedding capacity of 60141 bits and PSNR of 49 dB and 48dB has been achieved considering 512 × 512 and 256 × 256 images respectively. It can be further improved by adding Quantum based QR codes for authentication purposes.

References

[1] K. L. Chung, Y. H. Huang, W. M. Yan and W. C. Teng, "Distortion reduction for histogram modification-based reversible data hiding," *Applied Mathematics and Computation,* vol. 218, no. 9, pp. 5819–5826, 2012.

[2] D. Chyuan, C. H. Hu and C. C. Chiu, "Steganalysis of histogram modification reversible data hiding scheme by histogram feature coding," *International Journal of Innovative Computing Information and Control,* vol. 7, pp. 6571–6583, 2011.

[3] J. P. Guedon and N. Normand, "Mojette transform: Applications for image analysis and coding," *Visual Communications and Image Processing,* vol. 3024, pp. 873–884, 2007.

[4] X. Gui, X. Li and B. Yang, "A high capacity reversible data hiding scheme based on generalized prediction-error expansion and adaptive embedding," *Signal Processing,* vol. 98, pp. 370–380, 2013.

[5] K. S. Kim, M. J. Lee, H. Y. Lee and H. K. Lee, "Reversible data hiding exploiting spatial correlation between sub-sampled images," *Pattern Recognition,* vol. 42, no. 11, pp. 3083–3096, 2009.

[6] A. Kingston and F. Autrusseau, "Lossless image compression via predictive coding of discrete Radon projections" *Signal Processing: Image Communication,* vol. 23, pp. 313–324, 2008.

[7] C. Liu and J. Guédon, "Finding all solutions of three-material image reconstruction problem," *Journal of South China University of Technology*, vol. 41, no. 7, pp. 114–119, 2013.

[8] P. Mohan Kumar and K. L. Shunmuganathan, "A reversible high embedding capacity data hiding technique for hiding secret data in images," *International Journal of Computer Science and Information Security,* vol. 7, pp. 109–115, 2010.

[9] S. Maria Celestin Vigila and K. Muneeswaran, "Hiding of confidential data in spatial domain images using image interpolation," *International Journal of Network Security,* vol. 17, pp. 722–727, 2015.

[10] Z. Ni, Y. Q. Shi, N. Ansari and W. Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 16, no. 3, pp. 354–362, 2006.

[11] N. Normand, A. Kingston and P. Évenou, "A geometry driven reconstruction algorithm for the mojette transform," in *Discrete Geometry for Computer Imagery,* LNCS 4245, pp. 122–133, Springer, 2006.

[12] M. Nosrati, R. Karimi and M. Hariri, "Reversible data hiding: Principles, techniques, and recent studies," *World Applied Programming,* vol. 2, no. 5, pp. 349–353, 2012.

[13] A. Nag, S. Biswas, D. Sarkar and P. P. Sarkar, "Semi random position based steganography for resisting statistical steganalysis," *International Journal of Network Security,* vol. 17 pp. 57–65, 2015.

[14] H. T. Panduranga, H. S. Sharath Kumar and S. K. Naveen Kumar, "Hybrid approach for dual image encryption using nibble exchange and Hill-cipher," in *IEEE International Conference on Machine Vision and Image Processing,* pp. 101–104, 2012.

[15] L. Y. Por, D. Beh, T. F. Ang and S. Y. Ong, "An enhanced mechanism for image steganography using sequential colour cycle Algorithm," *International Arab Journal of Information Technology,* vol. 10, pp. 51–60, 2013.

[16] W. Puech, M. Chaumont and O. Strauss, "A reversible data hiding method for encrypted images," in *Processing of SPIE, Electronic Imaging, Security,*

Forensics, Steganography, and Watermarking of Multimedia, pp. 1–9, 2008.

[17] D. Rajendra, D. Kanphade and N. S. Narawade, "Forward modified histogram shifting based reversible watermarking with reduced pixel shifting and high embedding capacity," *International Journal of Electronics and Communication Engineering,* vol. 5, pp. 185–191, 2012.

[18] R. Ramaswamy and V. Arumugam, "Lossless data hiding based on histogram modification," *International Arab Journal of Information Technology,* vol. 9, no. 5, pp. 445–451, 2012.

[19] P. D. Ratna Raju, B. A. David and K. Prasada Rao, "Binary tree approach for data hiding based on histogram modification," *International Journal of Computer Applications,* vol. 5, pp. 21–24, 2010.

[20] B. Recur, H. Der Sarkissian, M. Servires, N. Normand and J. Guédon, "Validation of Mojette reconstruction from Radon acquisitions," in *IEEE International Conference on Image Processing,* pp. 1041–1045, 2013.

[21] E. Satir and H. A. Isik, "Huffman compression based text steganography method," *Multimedia Tools and Applications,* vol. 70, no. 3, pp. 2085–2110, 2014.

[22] I. Svalbe, A. Kingston, J. Guédon, N. Normand and S. Chandra, "Direct inversion of Mojette projections," in *IEEE International Conference on Image Processing,* pp. 1036–1040, 2013.

[23] S. A. Seyyedi, V. Sadau and N. Ivanov, "A secure steganography method based on integer lifting wavelet transform," *International Journal of Network Security,* vol. 18, pp. 124–132, 2016.

[24] W. L. Tai, C. M. Yeh and C. C. Chang, "Reversible data hiding based on histogram modification of pixel differences," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 19, pp. 906–910, 2009.

[25] Z. H. Wang, C. C. Chang, M. L. Li and Y. S. Cui, "Multi-dimensional and multi-level histogram-shifting-imitated reversible data hiding scheme," *Advances in Intelligent Systems and Applications,* vol. 21, pp. 149–158, 2013.

[26] Z. Zhao, H. Luo, Z. M. Lu and J. S Pan, "Reversible data hiding based on multilevel histogram modification and sequential recovery," *AEU-International Journal of Electronics and Communications,* vol. 65, no. 10, pp. 814–826, 2011.

Padmapriya Praveenkumar received her B.E (ECE) from Angala Amman college of Engineering and Technology and M.E (Communication system) from Jayaram college of Engineering and Technology. Currently she is working as an Assistant Professor III in the Department of ECE in SASTRA University, Thanjavur. She has a teaching experience of 13 years and she has published 26 Research articles in National & International journals. She is currently working towards her Ph.D. Degree in SASTRA University. Her research area includes Wireless communication and Steganography

K. Thenmozhi received her B.E (ECE) and M.E (Communication system) degrees from Regional Engineering college (NIT) Tiruchirappalli and Ph.D. from SASTRA University, Thanjavur. Currently she is working as an Associate Dean in the Department of ECE in SASTRA University, Thanjavur. She has a teaching experience of 20 years. Her current research area includes Wireless communication, Steganography and Information Theory and Coding. She has supervised more than 100 UG projects, 10 Master Students and Supervising 4 Ph.D. Scholars. So far she has published 53 Research articles in National & International journals@conferences. She received EDI award from broadcast Engineering Society for the year 2007.

John Bosco Balaguru Rayappan was born in Trichy, Tamil Nadu province, India in 1974. He received the B.Sc., M.Sc. and M.Phil. Degree in Physics from St. Joseph College, Bharathidasan University, Trichy and Ph.D. in Physics from Bharathidasan University, Trichy, Tamil Nadu India in 1994, 1996, 1998 and 2003, respectively. He joined the faculty of SASTRA University, Thanjavur, India in Dec 2003 and is now working as Professor & Associate Dean Research School of Electrical and Electronics Engineering at SASTRA University, Thanjavur, Tamil Nadu, India. His research interests include Lattice Dynamics, Nanosensors, Embedded System and Steganography. So far he has published 161+ Research articles in National and International journals and 14 conference papers. He has Supervised 25 Master Students and Supervising 5 Ph.D. Scholars. Currently he is working on four funded projects in the fields of Nanosensors and Steganography supported by DST and DRDO, Government of India, New Delhi. Indo-Swedish collaboration work.

R. Amirtharajan was born in Thanjavur, Tamil Nadu province India, in 1975. He received B.E. degree in Electronics and Communication Engineering from P.S.G. College of Technology, Bharathiyar University, Coimbatore, India in 1997. M.Tech. and Ph. D. from SASTRA University Thanjavur, India in 2007 and 2012 respectively. He joined SASTRA University, Thanjavur, Tamil Nadu, India (Previously Shanmugha College of Engineering) as a Lecturer in the Department of Electronics and Communication Engineering since 1997 and is now Associate Professor, His research interests include Image Processing, Information Hiding, Computer Communication and Network Security. So far, he filed one international patent; he has published more than 116+ research articles in national and international journals and 22 IEEE conference papers with 4 Best Paper Awards. He also holds the Certificate of Appreciation from IBM in 2009 for Great Mind Challenge, Mentor IBM Academic Initiative Program. Recently, he received the Founder Chancellor Award for the best Ph.D. thesis for 2013 from SASTRA University and he received the

SASTRA Anukul Puraskar for Higher Involvement in Research and Education Award for 2011-2012 and 2013-2014. He serves as a Life Member in CRSI, SSI, IAENG, and IACSIT. He also served as the TPC Member and Review Member for more than 30+ IEEE and Springer supported international conferences apart from more than 10 peer reviewed journals. He had been working on funded project in the field of steganography supported by DRDO, Government of India, New Delhi, India.

SMP: Scalable Multicast Protocol for Granting Authority in Heterogeneous Networks

Kuo-Jui Wei[1], Jung-San Lee[1], and Bo Li[2]

(Corresponding author: Jung-San Lee)

Department of Information Engineering and Computer Science[1]
Feng Chia University, Taichung 40724, Taiwan, R.O.C.
Department of Electrical Engineering and Computer Science[2]
Vanderbilt University, Nashville, Tennessee, USA
(Email: leejs@fcu.edu.tw)

(Received Aug. 6, 2015; revised and accepted Jan. 23 & Feb. 28, 2016)

Abstract

The fundamental function of the network protocol is to provide confidential communications or services for authorized participants over an insecure network. The proliferation of the Internet and mobile computing technologies, however, has led to emerging applications such as message services, pay-per-view, teleconference, and collaboration tasks. Especially, users now can surf over the Internet or get online for communications via wired, wireless, 3G, or LTE (Long Term Evolution) networks. Traditional peer-to-peer transmission protocol will no longer suffice for these types of applications. Consequently, point-to-group and group-to-group transmission have become important areas of focus. The main challenge in designing a secure multicast mechanism results from large groups and frequent key updates caused by members joining and leaving. To mitigate the encumbrance of group high-mobility in heterogeneous networks, we propose a subgroup-based multicast protocol adopting Lagrange Interpolating Polynomial technique. Simulation results show that the scalable multicast protocol (SMP) can not only preserve the forward and backward secrecy of group communications but also perform better than related works on system communication cost and storage consumption.

Keywords: Communication security, multicast, scalability

1 Introduction

Engineers have proposed many security protocols for providing confidential communications in large network groups; protocols for multicast communications are regarded as the most critical. The development of the Internet and mobile computing technologies has given rise to emerging applications such as teleconference, pay-TV, collaborating tasks, and message services [7, 9, 10, 12, 25, 29, 30]. Before obtaining the access to these services, resource providers have to delegate authority to legal subscribers. Hereafter, users can surf over the Internet or get online for communications via wired, wireless, 3G, or LTE networks. Traditional peer-to-peer communications do not suffice for these applications any more. Along with this trend, how to design high-performance peer-to-group and group-to-group communications has become an important research issue in heterogeneous networks [1, 4, 8, 11, 13, 15, 16, 17, 20, 22, 24, 26].

There are two main issues in designing a multicast mechanism: scalability and mobility. Scalability involves how to maintain the high performance in a large network for emerging applications, while mobility addresses how to efficiently complete key updates caused by frequent entrances and exits of members. As the fast development of networks has deeply affected the current world, more individuals are involved in this area with high frequency of mobility. Therefore, efficient solving plans of these two concerns are sure to contribute much to the network communications. Owing to the past researches, three main solutions have been proposed for providing secure multicast communications and key distribution: central control, distributed control, and subgroup control.

Central control: A central manager takes responsibility for the security of the entire group and key distribution. However, this solution is unsuitable for large networks, since the efficiency of the central manager will become the performance bottleneck of group communications. The failure of the central manager may lead to the inactive communication of the whole group [18, 31].

Distributed control: All group members take obligation for key generation and the security of the group. Since this solution is based on the Diffie-Hellman protocol, each group member must sustain exponential modulations for key updates caused by frequent members joining and leaving [5, 6, 14, 19] , which makes it infeasible for large networks.

Subgroup control: The group is divided into several subgroups, each of which is controlled by a subgroup manager. The scalability of this approach is better than the other solutions, because the failure of one subgroup manger does not result in inactive overall group communication [2, 21, 28].

Although subgroup control is more feasible for providing multicast communications in large and high-mobility networks, there are still several weaknesses within this approach. First, each group user must keep many secret keys, which is quite inconvenient for involved participants considering the key storage ability. For example, if the mobile users want to launch a teleconference referring to this multicast mechanism, then more keys are needed to keep, much harder it is for them to communicate efficiently. Second, when a member joins or leaves the group, both involved and non-involved participants must change their secret keys to confirm the forward and backward secrecy. Taking the growing scale of the network into account, under this system one node action of joining or leaving happens, all the nodes have to handle heavy computation burthen. It is clear that this is so inefficient and unsuitable for large scale networks with countless participants currently. What is more, as in this system each time all the nodes have to take some actions, the energy wasted is worth thinking about. Thus, on the purpose of improving the efficiency of the whole system and achieving the energy saving, how to lower down the number of nodes taking actions for every time is as worth as a key factor. That is to say, reducing the number of participants for these cases becomes a critical challenge in designing a feasible multicast framework.

In particular, the computation overhead of frequent key updates also becomes a heavy burden for group members in a high-mobility environment. Seeing to the quick growth of the work burden and high development of the life quality, it is more and more popular to move fast to deal with emergent issues without extra time delay. Therefore, this noble mobility character requires the fast and precious disposal of the key updating and session key reconstruction. At the same time, it is sure that the security of the key and message need to be guaranteed firmly without falling down corresponding to the reducing of dealing time.

Out of all the considerations mentioned above, this article proposes a novel subgroup-based multicast protocol (SMP) adopting Lagrange Interpolating Polynomial (LIP) technique, which not only preserves the functionality of subgroup control mechanisms but also mitigates the encumbrance of group high-mobility. Furthermore, each group user only needs to keep one secret key in his/her database, compared with the traditional ones, in which the individual key, key encryption key and group key have to be stored as least. In addition, the number of participants involved in member joining and leaving can be effectively reduced to decrease the whole communication time and save power, making it portable for large scale networks. Moreover, owing to the highly efficient algorithms for updating and reconstructing keys when there joins a new node or leaves an existent node, the time consumption during these processes is considerable cut down. In this way, high-mobility network environment can refer to this efficient mechanism to attain flexible usages. Besides, simulation results will demonstrate that SMP outperforms other related mechanisms in scalable as well as high-mobility network environments.

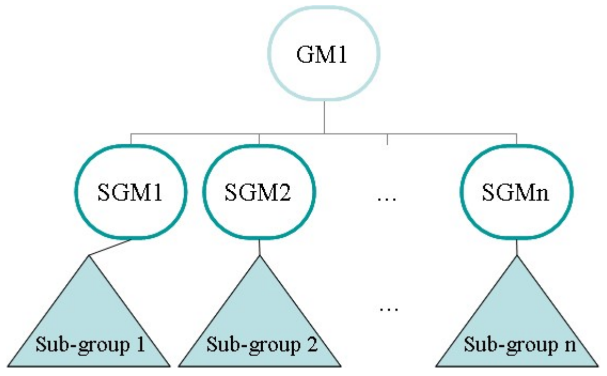


Figure 1: The structure of subgroup-based multicast

The rest of this paper is organized as follows. A model of subgroup-based multicast is described in Section 2, followed by the description of SMP in Section 3. Analyses of SMP are given in Section 4. Discussions and comparisons between other related works and SMP are shown in Section 5. Finally, we make conclusions in Section 6.

2 Model of Subgroup-based Multicast

The general structure of subgroup-based multicast protocols is illustrated in Figure 1. The main idea of these protocols is to divide the whole group into several subgroups. Each subgroup i is formed with a hierarchy structure and is controlled by a subgroup manager SGM_i, where $i = 1, 2, \cdots, n$ and n is the number of subgroups. The group manager GM shares a different secret key $K(GS_i)$ with each SGM_i and generates another secret key $K(GS)$ shared among all SGM_i's.

As shown in Figure 2, each internal node of subgroup i is a virtual node with a unique secret key and each leaf node denotes a subgroup member. Each member owns a private key and has to learn secret keys of the internal node on the path from the subgroup manager to himself/herself. For example, in Subgroup i, the user U_1 must know $K_i(h, 1)$, $K_i(h-1, 1)$, $\cdots$, $K_i(0, 1)$, where h is the height of subgroup i, and d is the maximum degree of each internal nodes.

Furthermore, several assumptions are made in the multicast system. First, when a new member wants to join

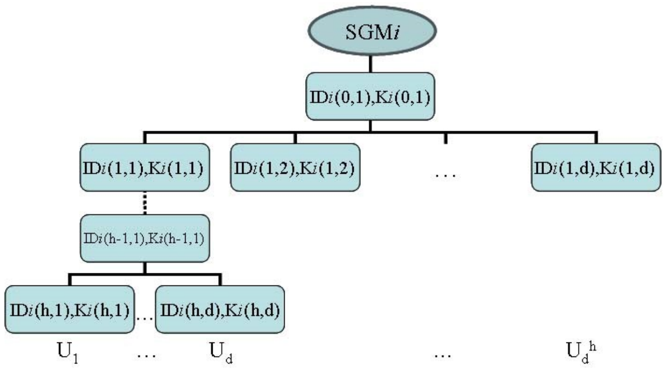


Figure 2: The hierarchy structure of Subgroup i

the group, GM must take responsibility for finding an empty place and generating a secret key for him/her. If all subgroups are full, GM has to create a new subgroup. Second, all nodes belonging to SGM_i are assumed to be trustworthy.

What is more, the subgroup managers ought to preserve the forward and backward secrecy of group communications to enhance the security of the system.

The Forward Secrecy: If a new member is permitted to join a subgroup, secret keys of the internal node on the path from the subgroup manager to himself/herself must be changed to prevent previous group messages from being learned by the new user.

The Backward Secrecy: In case that a member is expelled from the group, the subgroup manager has to modify secret keys of the internal node on the path from the node to its subgroup manager to stop the expellee from learning incoming group messages.

3 The Scalable Multicast Protocol (SMP)

Lagrange Interpolating Polynomial and a bulletin board are adopted in SMP to reduce the number of participants involved in member joining and leaving operations. Note that only GM and the legal SGM_i's can modify and update the bulletin board. The whole group is divided into several subgroups formed with hierarchy structures of height h. As illustrated in Figure 2, every node in the hierarchy structure of subgroup i is assigned a unique identity $ID_i(b,j)$ and a secret key $K_i(b,j)$, where $b = 0,1,\cdots,h$, $j = 1,2,\cdots,d$, and d is the maximum degree of the internal node in the hierarchy tree.

Before describing the broadcasting procedure of SMP, we first introduce the definition of Lagrange Interpolating Polynomial and how GM constructs the bulletin broad in Subsections 3.1 and 3.2.

3.1 Definition of Lagrange Interpolating Polynomial (LIP)

Let (x_1,y_1), (x_2,y_2), $\cdots$, (x_t,y_t) be t points on two-dimensional plane [21], N be a 128-bit prime, and a_0, a_1, $\cdots$, a_{t-1} be integers ranged within $[1,N\text{-}1]$. To attain a polynomial $f(x)$, where $y{=}f(x)$ passes through the t points, we refer to the Lagrange Polynomial to calculate

$$\begin{aligned} f(x) &= \sum_{j=1}^{t} y_j \prod_{i=1, i\neq j}^{t} \left(\frac{x-x_i}{x_j-x_i}\right) mod\ N \\ &= a_0 + a_1x + a_2x^2 + ... + a_{t-1}x^{t-1} mod\ N. \end{aligned}$$

Note that $y = f(x)$.

3.2 The Bulletin Board Setup

Here, we describe how GM constructs the bulletin board as shown in Table 1. For each internal node $ID_i(b,j)$ in subgroup i, SGM_i bottom-up computes a corresponding polynomial $ID_i(b,j)$_P(x) as follows, where $b = h\text{-}1, h\text{-}2, \cdots, 0$ and $j = 1,2,\cdots,d$.

Step 1: Computes d distinct hash values

$$\begin{aligned} h_{ib_1} &= h(K_i(b+1,1), ID_i(b,j), ID_i(b+1,2), \\ &\quad \cdots, ID_i(b+1,d)), \\ h_{ib_2} &= h(K_i(b+1,2), ID_i(b+1,1), ID_i(b,j), \\ &\quad \cdots, ID_i(b+1,d)), \\ &\vdots \qquad\qquad \vdots \\ h_{ib_d} &= h(K_i(b+1,d), ID_i(b+1,1), \\ &\quad ID_i(b+1,2), ID_i(b+1,3), \cdots, \\ &\quad ID_i(b+1,d-1), ID_i(b,j)). \end{aligned}$$

Step 2: Performs Lagrange Interpolating Polynomial on these coordinates $(h_{ib_1}$, $K_i(b,j) +$ $h_{ib_1})$, $(h_{ib_2}$, $K_i(b,j) + h_{ib_2})$, $\cdots$, and $(h_{ib_d}$, $K_i(b,j) + h_{ib_d})$, to obtain the polynomial

$$ID_i(b,j)\_P(x) = a_0 + a_1x + a_2x^2 + ... + a_{d-1}x^{d-1} \text{ mod } N,$$

where $a_0, a_1, \cdots, a_{d-1}$ are integers and $h(\cdot)$ is the secure one-way hash function.

Step 3: Publishes all identities of nodes and their corresponding polynomials on the bulletin board.

3.3 Message Broadcast Operation

While a message M needs to be broadcasted, GM randomly generates a new secret key $K(msg)$ to encrypt M. Next, GM computes the followings, $E_{K(msg)}[M]$ and $E_{K(GS)}[K(msg)]$, where $E_K[\cdot]$ is the AES-based symmetric encryption with secret key K. Then GM broadcasts the computation results to all SGM_i's. After receiving the messages, each SGM_i computes $K(msg) =$

Table 1: The example of the bulletin board

SGM_i

Nodes	Polynomiials
ID$_i(b,1)$	ID$_i(b,1)$_P(x)
ID$_i(b,2)$	ID$_i(b,2)$_P(x)
⋮	⋮
ID$_i(b$,d)	ID$_i(b$, d)_P(x)

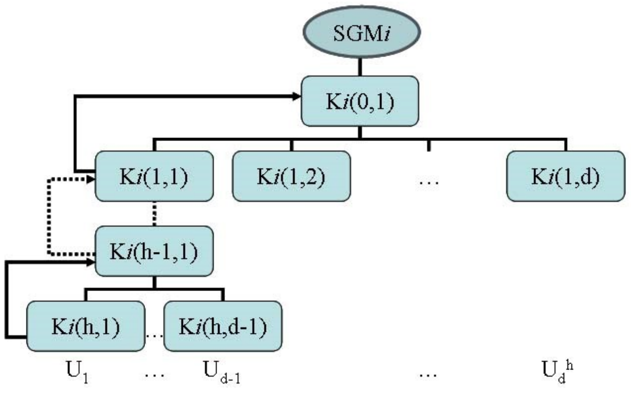


Figure 3: The example of message broadcast operation in subgroup i

$D_{K(GS)}[E_{K(GS)}[K(msg)]]$ and $E_{K_{i(0,1)}}[K(msg)]$, $D_K[\cdot]$ is the AES-based symmetric decryption with secret key K and $K_{i(0,1)}$ is the common secret key shared by all members in subgroup i. SGM_i then broadcasts the following messages to all subgroup users, $E_{K_{i(0,1)}}[K(msg)]$ and $E_{K(msg)}[M]$.

As shown in Figure 3, the subgroup member U_1 uses his/her secret key to obtain the secret key of the upper level by computing $K_i(h-1,1) = ID_i(h-1,1)\_P(h_{ib_1}) - h_{ib_1}$, where $h_{ib_1} = h(K_i(h,1), ID_i(h-1,1), ID_i(h,2), ID_i(h,3), \cdots, ID_i(h,d-1))$ is pre-computed by U_1 and b = h-1. By the same way, U_1 can quickly obtain $K_i(0,1)$ to retrieve $K(msg)$ and decrypt the message M.

4 Analyses of SMP

The previous section describes the normal operation of broadcasting a group message in static. The proliferation of the Internet and mobile computing technologies, however, makes the membership of a group vary from minute to minute. Therefore, we analyze how SMP manipulates the changes of membership and network topologies, which lead to much more frequent key update. Here in SMP, we mainly consider three types of mobility: subgroup manager joining, member joining, and member leaving.

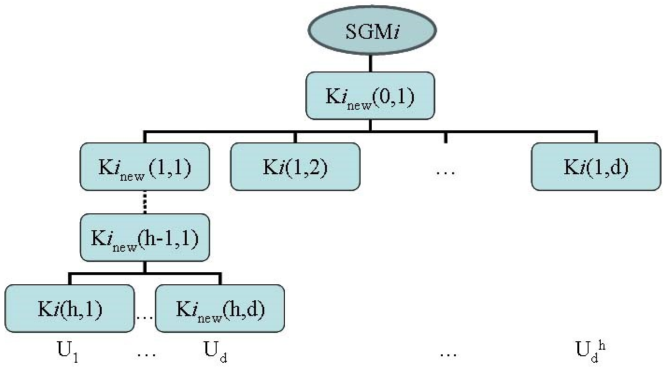


Figure 4: The example of member join operation in SMP

4.1 Subgroup Manager Joining Operation

If the scale of system users exceeds in the size of whole group, GM has to designate a new subgroup manager SGM_{n+1} and change the secret key from $K(GS)$ to $K_{new}(GS)$. Besides, GM has to generate a new secret key $K(GS_{n+1})$ shared between GM and SGM_{n+1}. Next, GM computes $E_{K(GS)}[K_{\text{new}}(GS)]$ and $E_{K(GS_{n+1})}[K_{new}(GS)]$.

GM then broadcasts the computation results to all SGM_i's including the new one. While receiving the messages, the original SGM_i's retrieve the new secret key $K_{new}(GS)$ by computing $D_{K(GS)}[E_{K(GS)}[K_{new}(GS)]]$.

On the other hand, the new subgroup manager retrieves the new secret key $K_{new}(GS)$ by calculating $D_{K(GS_{n+1})}[E_{K(GS_{n+1})}[K_{new}(GS)]]$.

Hence, the joining operation of a new subgroup manager is completed.

4.2 Member Joining Operation

While a new member U_d wants to join the communication group, GM has to find a suitable place and generate a secret key $K_{i_{new}}(h,d)$ for him/her. As illustrated in Figure 4, all secret keys of the path from SGM_i to U_d must be modified to preserve the forward secrecy. The secret key $K_i(b,1)$ must be changed, where $b = 0, 1, \cdots, h-1$. All involved internal nodes' polynomials published on the bulletin board will be updated by SGM_i. That is, SGM_i has to bottom-up perform Lagrange Interpolating Polynomial $(h-1)$ times to reconstruct $(h-1)$ involved polynomials.

For each involved internal node $ID_i(b,1)$, where $b = h-1, h-2, \cdots, 0$ (i.e. the internal nodes on the path from U_d to SGM_i), SGM_i executes the followings.

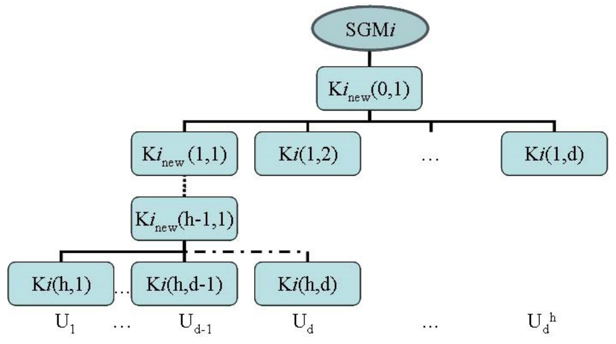


Figure 5: The example of member leave operation in SMP

Step 1: Computes d distinct hash values

$$\begin{aligned}
h_{ib_1} &= h(K_i(b+1,1), ID_i(b,1), ID_i(b+1,2), \\
&\quad \cdots, ID_i(b+1,d)), \\
h_{ib_2} &= h(K_i(b+1,2), ID_i(b+1,1), ID_i(b,1), \\
&\quad ID_i(b+1,3), ID_i(b+1,4), \\
&\quad \cdots, ID_i(b+1,d)), \\
&\vdots \qquad\qquad \vdots \\
h_{ib_{(d-1)}} &= h(K_i(b+1,d-1), ID_i(b+1,1), \\
&\quad ID_i(b+1,2), ID_i(b+1,3), \cdots, \\
&\quad ID_i(b+1,d-2), ID_i(b,1)), \\
&\quad ID_i(b-1,d)), \\
h_{ib_d} &= h(K_i(b+1,d), ID_i(b+1,1), \\
&\quad ID_i(b+1,2), ID_i(b+1,3), \cdots, \\
&\quad ID_i(b+1,d-1), ID_i(b,1)).
\end{aligned}$$

Step 2: Performs Lagrange Interpolating Polynomial on these coordinates (h_{ib_1}, $K_i(b,1) + h_{ib_1}$), (h_{ib_2}, $K_i(b,1) + h_{ib_2}$, $\cdots$, and (h_{ib_d}, $K_i(b,1) + h_{ib_d}$), to obtain the polynomial

$$ID_i(b,1)\_P(x) = a'_0 + a'_1 x + \cdots + a'_{d-1} x^{d-1} \bmod N,$$

where $a'_0, a'_1, \cdots, a'_{d-1}$ are integers.

Step 3: Updates the modified information on the bulletin board as shown in Table 1.

4.3 Member Leaving Operation

While a user U_d leaves Subgroup i, as illustrated in Figure 5, all secret keys of the path from SGM_i to U_d must be modified to preserve the backward secrecy. The secret key $K_i(b,1)$ must be changed, where $b = 0, 1, \cdots, h-1$. All involved internal nodes' polynomials published on the bulletin board have to be modified by SGM_i in time. That is, SGM_i needs to bottom-up perform Lagrange Interpolating Polynomial $(h-1)$ times to reconstruct $(h-1)$ involved polynomials.

For each involved internal node $ID_i(b,1)$, where $b = h-1, h-2, \cdots, 0$ (i.e. the internal nodes on the path from U_d to SGM_i), SGM_i executes the followings.

Step 1: Computes

$$\begin{aligned}
h_{ib_1} &= h(K_i(b+1,1), ID_i(b,1), ID_i(b+1,2), \\
&\quad \cdots, ID_i(b+1,d-1)) \\
h_{ib_2} &= h(K_i(b+1,2), ID_i(b+1,1), ID_i(b,1), \\
&\quad ID_i(b+1,3), ID_i(b+1,4), \cdots, \\
&\quad ID_i(b+1,d-1)) \\
&\vdots \qquad\qquad \vdots \\
h_{ib_{d-1}} &= h(K_i(b+1,d-1), ID_i(b+1,1), \\
&\quad ID_i(b+1,2), ID_i(b+1,3), \cdots, \\
&\quad ID_i(b+1,d-2), ID_i(b,1)).
\end{aligned}$$

Step 2: Performs Lagrange Interpolating Polynomial on these coordinates (h_{ib_1}, $K_{i_{new}}(b,1)+$ h_{ib_1}), (h_{ib_2}, $K_{i_{new}}(b,1)$ + h_{ib_2}, $\cdots$, and ($h_{ib_{(d-1)}}$, $K_{i_{new}}(b,1)$ + $h_{ib_{(d-1)}}$), to obtain the polynomial

$$\begin{aligned}
ID_i(b,1)\_P(x) &= a''_0 + a''_1 x + a''_2 x^2 + \cdots \\
&\quad + a''_{d-1} x^{d-1} \bmod N,
\end{aligned}$$

where $a''_0, a''_1, \cdots, a''_{d-1}$ are integers.

Step 3: Updates the modified information on the bulletin board.

5 Discussions

We discuss how SMP is able to preserve the forward and backward secrecy during group communications and confirm the broadcast of messages at length in this section. Furthermore, we also present the performance comparisons with related works of SMP to demonstrate its convenience, low computation cost, and low storage space.

5.1 Security Examination

Since the member joining operation and member leaving operation are always performed frequently in dynamic network groups, it is important to prevent messages from being illegally shared. The security of SMP is based on three cryptographic assumptions.

Secure AES-based symmetric en/decryption $[m]_k$.
With the message m, it is relatively easy to encrypt m as $[m]_k$ with an AES-based symmetric key k, while it is computationally infeasible to retrieve m from $[m]_k$ without the knowledge of k.

Discrete logarithm assumption.
Given a generator g, a large prime p, and a random number $x \in Z_p$, it is easy to compute $y = g^x \bmod p$. Then it is computationally infeasible to compute x just referring to y, g, and p. Note that x is called the discrete logarithm of y with respect to g.

Secure one-way hash function $h(\cdot)$.

1) Preimage resistance: As to a message m, it is easy to compute $h(m)$; nevertheless, it is computationally infeasible to attain m only from the knowledge of $h(m)$.
2) 2nd-preimage resistance: According to $h(m)$, it is impossible to find m' such that $h(m) = h(m')$ except applying the brute-force method.

5.1.1 Preserving the Forward Secrecy

As depicted in Figure 4, during the entrance of a new member U_d, he/she is assigned an identity $ID_{i_{new}}(h,d)$ and a secret key $K_{i_{new}}(h,d)$, which U_d can make use of to construct secret keys, $K_{i_{new}}(h-1,1)$, $K_{i_{new}}(h-2,1)$, $\cdots$, $K_{i_{new}}(0,1)$. To keep U_d from learning previous group messages, SMP must prevent U_d from figuring out the secret keys, $K_i(h-1,1)$, $K_i(h-2,1)$, $\cdots$, $K_i(0,1)$, shown in Figure 3. Here, we prove that SMP can confirm the forward secrecy by Proposition 1.

Proposition 1. *If a new member U_d wants to learn the previous messages shared between old group members, he/she must fail.*

Proof. To learn the advanced messages, U_d must first obtain one of the hash values h_{ib_1}, h_{ib_2}, $\cdots$, $h_{ib_{d-1}}$, where $b = h-1$. Then, U_d can apply the hash value to $ID_i(h-1,1)\_P(x)$ in order to retrieve $K_i(h-1,1)$. Given $K_{i_{new}}(h,d)$, $ID_i(h-1,1)$, $ID_i(h,2)$, $ID_i(h,3)$, $\cdots$, $ID_i(h,d-1)$, $ID_{i_{new}}(h,d)$, U_d can obtain the hash value,

$$\begin{aligned} h^*_{ib_d} &= h(K_{i_{new}}(h,d), ID_i(h,1), ID_i(h,2), ID_i(h,3), \\ &\quad \cdots, ID_i(h,d-1), ID_i(h-1,1)). \end{aligned}$$

Since

$$\begin{aligned} h_{ib_1} &= h(K_i(h,1), ID_i(h-1,1), ID_i(h,2), \\ &\quad ID_i(h,3), \cdots, ID_i(h,d-1)), \\ h_{ib_2} &= h(K_i(h,2), ID_i(h,1), ID_i(h-1,1), \\ &\quad ID_i(h,3), \cdots, ID_i(h,d-1)), \\ &\vdots \qquad \vdots \\ h_{ib_{d-1}} &= h(K_i(h,d-1), ID_i(h,1), ID_i(h,2), \cdots, \\ &\quad ID_i(h,d-2), ID_i(h-1,1)). \end{aligned}$$

We infer that $h^*_{ib_d}$ must be different from h_{ib_1}, h_{ib_2}, $\cdots$, and $h_{ib_{(d-1)}}$ under the assumption of the secure one-way hash function. That is, U_d cannot apply $K_{i_{new}}(h,d)$ and $ID_i(h-1,1)\_P(x)$ to obtain $K_i(h-1,1)$. Similarly, U_d is unable to learn $K_i(h-2,1)$, $K_i(h-3,1)$, $\cdots$, and $K_i(0,1)$.

Again, since N is a large prime, if U_d wants to resolve $ID_i(h-1,1)\_P(x)$ without one of the hash values h_{ib_1}, h_{ib_2}, $\cdots$, and $h_{ib_{d-1}}$, he/she must face the difficulty of solving the discrete logarithm problem. It is computationally infeasible for U_d to achieve this attempt. Furthermore, since we assume a secure AES-based symmetric en/decryption in the multicast system, U_d cannot compromise the previous group messages without $K_i(0,1)$. □

5.1.2 Preserving the Backward Secrecy

As illustrated in Figure 5, when U_d is expelled from subgroup i, SMP has to prevent U_d from listening to group communications continually. Consequently, SGM_i must modify the secret keys of the internal node on the path from SGM_i to U_d. That is, SGM_i must reconstruct secret keys $K_{i_{new}}(h-1,1)$, $K_{i_{new}}(h-2,1)$, $\cdots$, and $K_{i_{new}}(0,1)$ and then apply them to update $ID_i(h-1,1)\_P(x)$ on the public board. We demonstrate that SMP can confirm the backward secrecy by Proposition 2.

Proposition 2. *If an expellee U_d wants to learn the content of the incoming messages shared among current group members, he or she must fail.*

Proof. To uncover incoming group messages, U_d must firstly obtain one of the hash values

$$h^*_{ib_1}, h^*_{ib_2}, \cdots, h^*_{ib_{d-1}},$$

where $b = h-1$,

$$\begin{aligned} h^*_{ib_1} &= h(K_i(h,1), ID_i(h-1,1), ID_i(h,2), \\ &\quad ID_i(h,3), \cdots, ID_i(h,d-1)), \\ h^*_{ib_2} &= h(K_i(h,2), ID_i(h,1), ID_i(h-1,1), \\ &\quad ID_i(h,3), \cdots, ID_i(h,d-1)), \\ &\vdots \qquad \vdots \\ h^*_{ib_{d-1}} &= h(K_i(h,d-1), ID_i(h,1), ID_i(h,2), \cdots, \\ &\quad ID_i(h,d-2), ID_i(h-1,1)). \end{aligned}$$

It is clear that SGM_i does not use h_{ib_d} and $K_i(h,d)$ to update $ID_i(h-1,1)\_P(x)$. Hence, it is computationally infeasible for U_d to solve $ID_i(h-1,1)\_P(x)$ without the correct hash values, under the assumption of the discrete logarithm. Furthermore, it is also computationally infeasible to compute a hash value that equals to $h^*_{ib_1}$, $h^*_{ib_2}$, $\cdots$, or $h^*_{ib_{d-1}}$ under the assumption of the secure one-way hash function. Since we assume a secure AES-based symmetric en/decryption in multicast systems, U_d cannot listen to incoming group messages without knowing $K_{i_{new}}(0,1)$. □

5.2 Performance Evaluations and Comparisons

In the following subsection, we compare SMP with related works to show its outstanding advantages. Notations used in Table 2 are defined as follows.

n: total number of subgroups;

m: total number of subgroup members;

d: maximum degree of each internal node in the hierarchy structure;

h: height of each subgroup ($m = d^h$);

$h_1 : log_d(n \times m)$;

Table 2: Comparisons with related works

		[12]	[2]	[31]	[21]	SMP
Member join	SGM	$2(h+1)$K	$2h$K	$2h_1$K	2K	(h-1)P+(hd)H
	IN	$(h+1)$K	hK	h_1K	1K	0
	NON	$(d/d-1)$K	$(d/d-1)$K	$(d/d-1)$K	1K	0
Member leave	SGM	$[2(hd)+(h-d)]$K	$[2(hd-1)+(h-d-1)]$K	$[2(h_1d-1)+(h_1-d-1)]$K	$(m-1)$K	(h-1)P+(hd)H
	IN	-	-	-	-	-
	NON	$(3d/d-1)$K	$(3d/d-1)$K	$(3d/d-1)$K	1K	0
Message broadcast	SGM	2K	2K	2K	2K	2K
	IN	1K	1K	1K	1K	hY+1K
	NON	-	-	-	-	-

IN: involved member;

NON: non-involved member;

K: computation overhead of a symmetric en/decryption operation;

H: computation overhead of a one-way hash operation;

P: computation overhead of constructing the polynomial by LIP;

Y: computation overhead of obtaining y with input x, where $y = f(x)$;

To access broadcast messages and update subgroup session keys, both involved and non-involved members must perform cryptographic operations frequently within the system. Excluding the security consideration, whether it is able to achieve these operations efficiently often dominates the evaluation of a multicast mechanism. As illustrated in Table 2, SMP adopts the Lagrange Interpolating Polynomial and the secure one-way hash function in addition to the symmetric en/decryption algorithm.

As mentioned in [28], a one-way hash function can be executed at least 10 times faster than a symmetric en/decryption. It is clear that the computation overhead of constructing the polynomial by LIP is quite lighter than that of performing the symmetric en/decryption, since the construction of LIP polynomial is based on simple multiplication, while the en/decryption function needs round operations [3, 6, 27]. Hence, SMP outperforms other works in terms of member joining and member leaving operations.

Specifically, when a member joins or leaves the subgroup, all subgroup members must update session keys to confirm the forward and backward secrecy as presented in [2, 12, 21, 31]. In SMP, by contrast, only the SGM has to construct and update corresponding polynomials on the bulletin board. This can effectively lower down the computation overhead of group members. To preserve this benefit, involved users must perform $h^*Y + 1K$ operations to retrieve a broadcast message; users in other works only need to execute $1K$ operation. The simulation results can show that this extra overhead is still acceptable.

5.3 Simulation Results

As is shown, simulation results demonstrate the superiority of SMP over its predecessors. Simulators were executed in the VC6.0 language. Cryptographic routines, including the AES algorithm and SHA-1, were implemented using the public OpenSSL library [23]. Note that the processing time of the simulation does not include the time used to authenticate the new member. That is, only the computation overhead of obtaining broadcasted message and constructing session keys is taken into consideration in the simulation. The symmetric en/decryption algorithm and the secure one-way hash function used in simulators are AES-128 and SHA-1, respectively. The encrypted message is 512k bytes, and the large prime N adopted in LIP is 128 bits. The ratio of member joining and member leaving to message broadcasting operation is set to 1:1:1.

To begin, we set the total number of subgroups (n to four) and the degree of each internal node in the hierarchy structure (d to two). The average SGM processing time (AvrST) is the mean of the total time needed for SGM to complete one member joining operation, one member leaving operation, and one message broadcasting operation. The average IN processing time (AvrIT) is the mean of the time consumption that takes involved nodes to complete one member joining operation and obtain one broadcasted message. The average NON processing time (AvrNT) is the mean of the time charge that takes a non-involved member to complete one member joining operation and one member leaving operation.

Table 3 compares SMP with related works in terms

Table 3: AvrST/AvrIT/AvrNT vs. m (512K bytes)

d=2, 512k		m(nodes)						
		64	128	256	512	1024	2048	4096
AvrST (seconds)	[12]	0.3128	0.3477	0.3826	0.4175	0.4524	0.4873	0.5222
	[2]	0.1947	0.2295	0.2648	0.2991	0.3341	0.3683	0.4037
	[31]	0.2643	0.2992	0.3339	0.3687	0.4032	0.4381	0.4729
	[21]	0.3338	0.6517	1.2872	2.5582	5.1002	10.1851	20.3541
	SMP	0.0181	0.0198	0.0223	0.0241	0.0254	0.0271	0.0287
AvrIT (seconds)	[12]	0.0538	0.0581	0.0622	0.0664	0.0705	0.0747	0.0789
	[2]	0.0352	0.0398	0.0451	0.0499	0.0559	0.0598	0.0657
	[31]	0.0459	0.0501	0.0559	0.0601	0.0657	0.0699	0.0758
	[21]	0.0101	0.0101	0.0101	0.0101	0.0101	0.0101	0.0101
	SMP	0.0062	0.0062	0.0063	0.0063	0.0064	0.0065	0.0065
AvrNT (seconds)	[12]	0.0099	0.0099	0.0099	0.0099	0.0099	0.0099	0.0099
	[2]	0.0397	0.0397	0.0397	0.0397	0.0397	0.0397	0.0397
	[31]	0.0397	0.0397	0.0397	0.0397	0.0397	0.0397	0.0397
	[21]	0.0099	0.0099	0.0099	0.0099	0.0099	0.0099	0.0099
	SMP	0	0	0	0	0	0	0

Table 4: AvrST/AvrIT/AvrNT vs. m (1M bytes)

d=2, 1M		m(nodes)						
		64	128	256	512	1024	2048	4096
AvrST (seconds)	[12]	0.6631	0.7316	0.8001	0.8686	0.9371	1.0056	1.0741
	[2]	0.3849	0.4537	0.5227	0.5914	0.6599	0.72791	0.7979
	[31]	0.5224	0.5914	0.6599	0.7287	0.7978	0.8665	0.9357
	[21]	0.6598	1.2897	2.5484	5.0668	10.1028	20.1754	40.3198
	SMP	0.0288	0.0299	0.0321	0.0337	0.0353	0.0367	0.0384
AvrIT (seconds)	[12]	0.0975	0.1076	0.1177	0.1278	0.1379	0.148	0.1581
	[2]	0.0697	0.0798	0.0895	0.0997	0.1094	0.1192	0.1289
	[31]	0.895	0.0997	0.1094	01192	0.1291	0.1388	0.1487
	[21]	0.0207	0.0207	0.0207	0.0207	0.0207	0.0207	0.0207
	SMP	0.0114	0.0114	0.0114	0.0114	0.0114	0.0114	0.0114
AvrNT (seconds)	[12]	0.0199	0.0199	0.0199	0.0199	0.0199	0.0199	0.0199
	[2]	0.0791	0.0791	0.0791	0.0791	0.0791	0.0791	0.0791
	[31]	0.0791	0.0791	0.0791	0.0791	0.0791	0.0791	0.0791
	[21]	0.0199	0.0199	0.0199	0.0199	0.0199	0.0199	0.0199
	SMP	0	0	0	0	0	0	0

of AvrST/AvrIT/AvrNT versus the number of nodes m. Furthermore, the input size of AES-128 in Table 3 is set to 512k bytes, while that in Table 4 is set to 1M bytes. Owing to this simulating method, it can be concluded that the security of the whole system is able to be enhanced without prolonging the time consumption for member joining/leaving and message broadcasting process. As is shown in Table 3, the time needed by SMP for AvrST is only 0.0181 seconds, while the lowest time cost of related works is still 0.1947 seconds under the condition that there are 64 nodes. Even though the time charge as AvrST grows in all theses methods along with the increase of nodes, SMP just needs 0.0287 seconds for 4096 nodes, while that is 0.4037 seconds at least in the related works. It is clear that SMP outperforms other approaches in all listed cases.

Although involved users in SMP must perform extra h*Y operations in order to retrieve a broadcast message, simulation results prove that the extra overhead is acceptable and still leave SMP the most optimal method among all the related multicast mechanisms. As is seen, it is the same with the AvrIT. In addition, all AvrNT values are zero in SMP, which confirms that in this method non-involved nodes do not need extra processing time in member joining/leaving operation. This advantage can effectively reduce the bandwidth consumption and computation overheads of non-involved nodes while maintaining high network mobility. As mentioned above, while in Table 4 the inputted size of AES-128 is set to 1M bytes, we can obtain the similar results. That is, no matter how the number of nodes rises, all the costs of SMP are the least in AvrST, AvrIT and AvrNT among related works. Since time depletion is the key factor for multicasting systems, this obvious advantage of time saving for SMP makes it more efficient in dynamic networks.

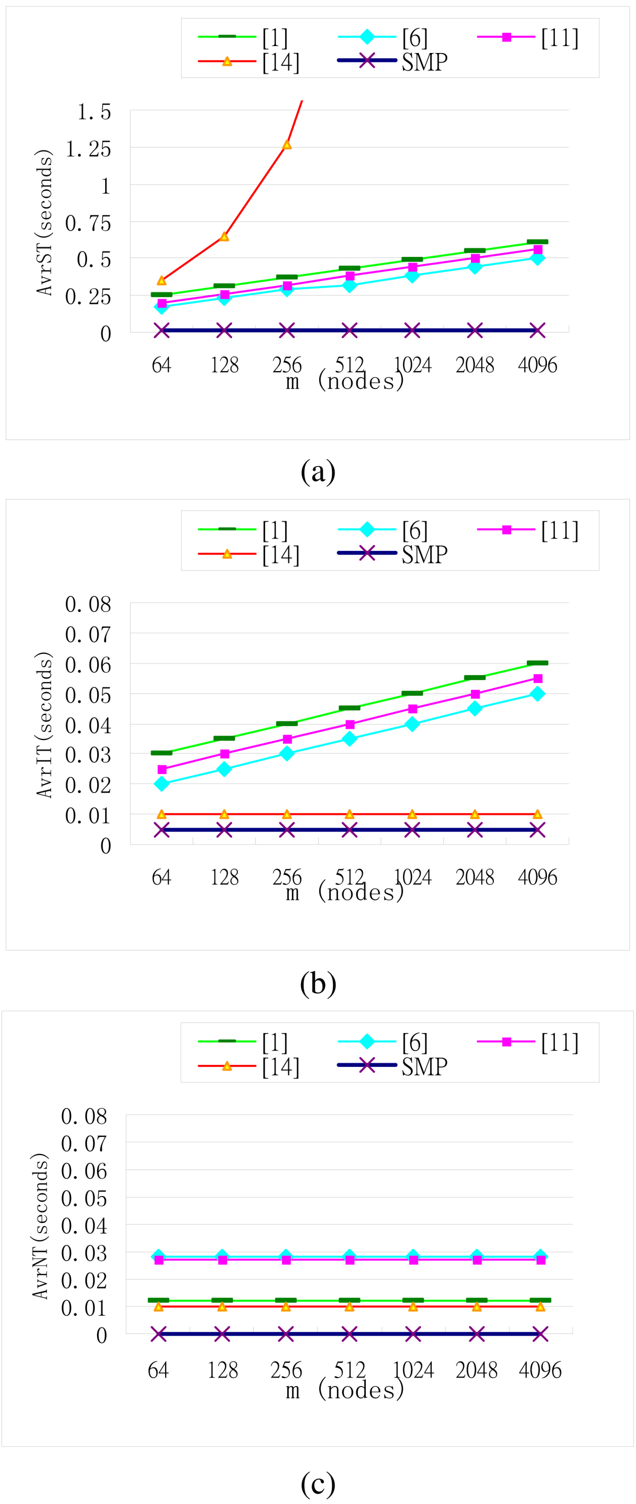
Besides, we adjust $d = 4$ to demonstrate the practicability of SMP in Figure 6. The inputted size of AES-128 in Figures 6(a), (b), and (c) is 512k bytes, while that in Figures 6(d), (e), and (f) is 1M bytes to show the security guaranteeing of SMP. As is displayed, Figure 6 shows the comparisons of SMP with related works in terms of AvrST/AvrIT/AvrNT versus m. From the trends shown in Figure 6, it is obvious that SMP still outperforms the works of [12, 2, 31, 21] considering the computation cost and communication cost, which are based on the time consumption. In Figure 6(a), the AvrST of SMP ranges from 0.0143 seconds to 0.0239 seconds; while in Figure 6(b), the AvrIT of SMP ranges from 0.0064 seconds to 0.0067 seconds.

Compared with the time consumption of related works [2, 12, 21, 31] , in which the lowest AvrST is 0.315 (second) and the lowest AvrIT is 0.0214 (second), it implies that SMP has more advantages under this condition. Moreover, as to the AvrNT, SMP costs nothing with m raging from 64 to 4096. That means, within SMP the non-involved nodes need not to perform any computation or related actions for updating session keys during the node joining/leaving process. In this way, the computation and communication are largely lowered down for the whole system. Furthermore, concerned by the insecurity within all kinds of networks, the encryption algorithm and the length of the key have to be cared more seriously. According to Figure 6(a) and Figure 6(d), along with the extension of the AES-128, the time consumption in SMP just grows 0.0213 seconds to 0.0317 seconds for AvrST, which rises a little. While, for the related works, this time charge is magnified on an obvious level. The similar situations appear in AvrIT and AvrNT, as shown in Figure 6(b) vs. Figure 6(e) and Figure 6(c) vs. Figure 6(f), which emphasize that SMP can hold with the extension of secret key to enhance the reality of the whole multicasting system.

As is shown in the simulation, SMP can effectively reduce the overhead of SGM in broadcasting messages and updating session keys. It can also reduce the overhead of involved nodes in joining a subgroup and obtaining broadcast messages. Furthermore, SMP can decrease the overhead of non-involved nodes in completing member joining/leaving operations.

In addition, taking the key storage cost into account, each group participant in SMP only needs to keep one secret key in the database compared with the traditional methods, where at least the individual key, key encryption key and group key have to be stored. The pertinent comparisons of SMP and other works [2, 12, 21, 31] versus m are shown in Figure 7. Here we set the degree of each internal node in the hierarchy structure, d=4. It can be seen that participants in SMP are able to store only their secret keys to complete member joining, member leaving as well as message broadcasting operations, while those in others still have to store more extra keys. Here we evaluate the key storage cost of SMP and related works basing on [12]. As [12] proposes an optimal key tree structure to reduce the storage burden in traditional methods, we set it as the base to display the times that each node in other related works and SMP has of it, called the Times of Key Storage (ToKS).

Figure 7 shows that related words all have several times of [12] in ToKS. And together with the increasing number of nodes from 64 to 4096, the key storage burden of related works grows almost 3 times. On the contrary, thanks to the LIP and subgroup mechanism, SMP achieves a much lower rate of [12] when nodes ranging from 64 to 4096. What is more, as in SMP each participant has to keep only one secret key, the key storage of users stays static no matter how many nodes there are. Therefore, as is displayed in Figure 7, SMP is able to save more space on the key storage management, which character is not affected by the number of participants. As is known to all, it can never be ignored to simultaneously reduce the overhead of communication cost as well as that of the key storage cost. Therefore, by successfully reducing the key storage cost, SMP outperforms other related mechanisms as the simulation results demonstrated.



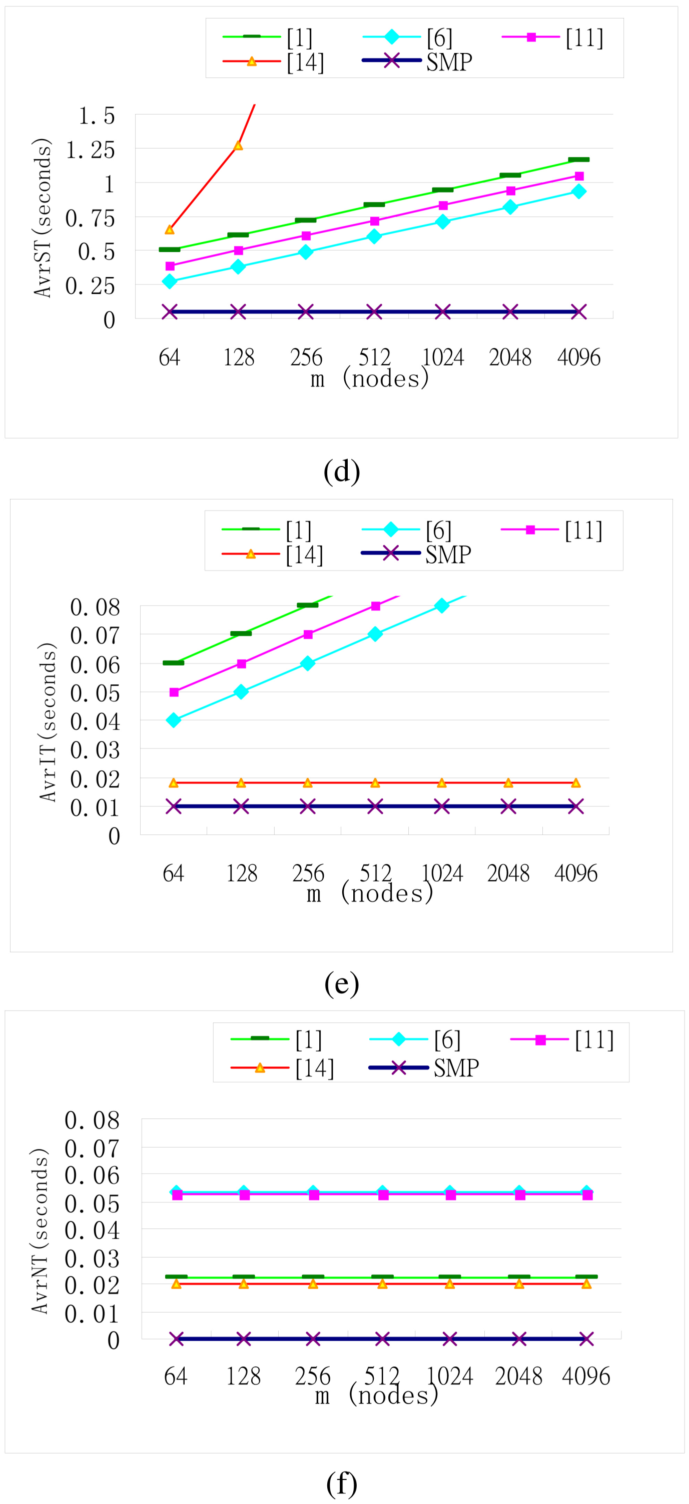


Figure 6: AvrST/AvrIT/AvrNT vs. $m, d = 4$

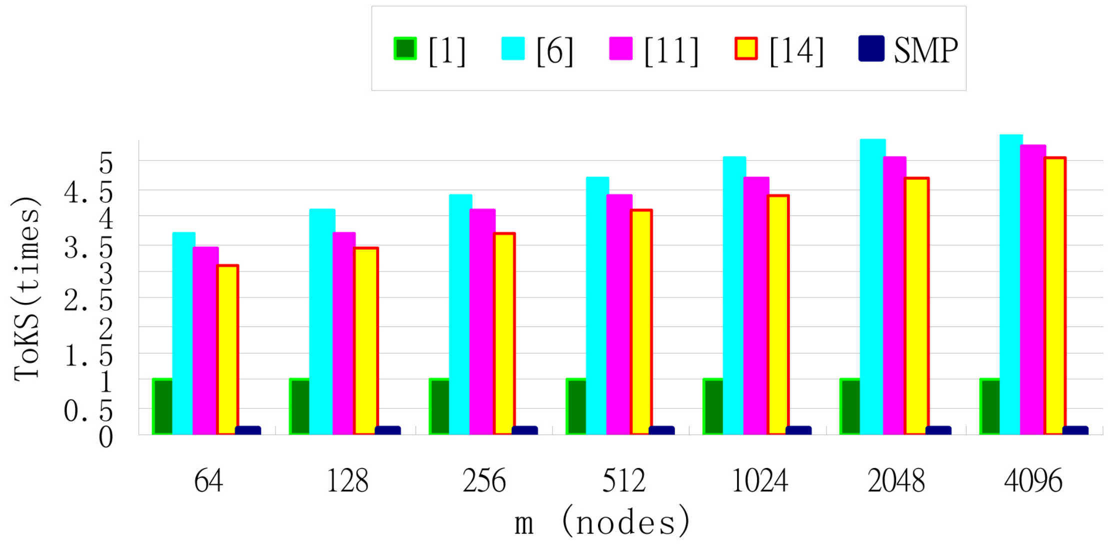


Figure 7: Key storage cost vs. $m, d = 4$

6 Conclusions

On all accounts, the subgroup control solution is more feasible for granting authority in large and high-mobility networks. However, we find that each subgroup member in the multicast system must keep many secret keys in his/her database. Furthermore, when a member joins or leaves the group, involved participants must modify their secret keys to preserve the forward and backward secrecy. In SMP, these disadvantages can be effectively improved by LIP technique. Specifically, simulation results have shown that SMP outperforms other works in terms of broadcasting messages and updating session keys to reduce communication cost, computation cost, and storage space. Therefore, SMP is more suitable for being applied to a large dynamic network environment.

References

[1] D. S. AbdElminaam, H. M. A. Kader, M. M. Hadhoud, and S. M. El-Sayedr, "Increase the performance of mobile smartphones using partition and migration of mobile applications to cloud computing," *International Journal of Electronics and Information Engineering*, vol. 1, no. 1, pp. 34–44, 2014.

[2] H. K. Aslan, "A scalable and distributed multicast security protocol using a subgroup-key hierarchy," *Computers and Security*, vol. 23, pp. 320–329, 2004.

[3] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," *Journal of Cryptology*, vol. 4, no. 1, pp. 3–72, 1991.

[4] C. Campolo, C. Casetti, C.F. Chiasserini, and A. Molinaro, "A multirate MAC protocol for reliable multicast in multihop wireless networks," *Computer Networks*, vol. 56, no. 5, pp. 1554–1567, 2012.

[5] T. Y. Chang and M. S. Hwang, "User-anonymous and short-term conference key distribution system via link-layer routing in mobile communications," *International Journal of Mobile Communications*, vol. 9, no. 2, pp. 144–158, 2011.

[6] D. W. Davies, "Some regular properties of DES," in *Advances in Cryptology (Crypto'82)*, pp. 89–96, Springer, 1983.

[7] G. G. Deverajan and R. Saravanan, "A novel trust based system to detect the intrusive behavior in MANET," *International Journal of Electronics and Information Engineering*, vol. 3, no. 1, pp. 31–43, 2015.

[8] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, pp. 644–654, 1976.

[9] S. M. El-Sayed, H. M. A. Kader, M. M. Hadhoud, and D. S. AbdElminaam, "Mobile cloud computing framework for elastic partitioned/modularized applications mobility," *International Journal of Electronics and Information Engineering*, vol. 1, no. 2, pp. 53–63, 2014.

[10] D. He, C. Chen, M. Ma, S. Chan, and J. Bu, "A secure and efficient password-authenticated group key exchange protocol for mobile ad hoc networks," *International Journal of Communication Systems*, vol. 26, no.4, pp. 495–504, 2013.

[11] I. Ingemarsson, D. Tang, and C. Wong, "A conference key distribution system," *IEEE Transactions on Information Theory*, vol. 28, no. 5, pp. 714–720, 1982.

[12] D. H. Je and S. W. Seo, "New key tree management protocol for the efficiency of storage and computation time in secure multicast communication," in *Proceedings of Wireless VITAE*, pp. 707–711, Aalbor, Denmark, 2009.

[13] G. Kadir, T. Kuseler, and I. A. Lami, "SMPR: a smartphone based MANET using prime numbers to enhance the network-nodes reachability and security of

routing protocols," *International Journal of Network Security*, vol. 18, no. 3, pp. 579–589, 2016.

[14] A. Kumar and S. Tripathi, "Anonymous ID-based Group Key Agreement Protocol without Pairing," *International Journal of Network Security*, vol. 18, no. 2, pp. 263–273, 2016.

[15] J. S. Lee, P. Y. Lin, and C. C. Chang, "Lightweight secure roaming mechanism between GPRS/UMTS and wireless LANs," *Wireless Personal Communications*, pp. 569–580, 2010

[16] G. Li, Q. Jiang, Y. Shi, and F. Wei, "Anonymous network information acquirement protocol for mobile users in heterogeneous wireless networks," *International Journal of Network Security*, vol. 18, no. 1, pp. 193–200, 2016.

[17] Y. Li and I. R. Chen, "Hierarchical agent-based secure and reliable multicast in wireless mesh networks, " *Computer Communications*, vol. 36, no. 14, pp. 1515–1526, 2013.

[18] Y. Li and I. R. Chen, "Dynamic agent-based hierarchical multicast for wireless mesh networks,," *Ad Hoc Networks*, vol. 11, no. 6, pp. 1683–1698, 2013.

[19] Z. Li, C. Wang, C. Jiang, and X. Li, "Multicast capacity scaling for inhomogeneous mobile ad hoc networks," *Ad Hoc Networks*, vol. 11, no. 1, pp. 29–38, 2013.

[20] D. Manivannan and P. Neelamegam, "An efficient key management scheme in multi-tier and multi-cluster wireless sensor networks," *International Journal of Network Security*, vol. 17, no. 6, pp. 651–660, 2015.

[21] S. Mittra, "IOLUS: A framework for scalable secure multicasting," in *Proceedings of ACM SIGCOM*, pp. 277–278, Cannes, France, 1997.

[22] N. Omheni, F. Zarai, M. S. Obaidat, K.-F. Hsiao, and L. Kamoun, "A novel media independent handover-based approach for vertical handover over heterogeneous wireless networks," *International Journal of Communication Systems*, vol. 27, no. 5, pp. 811–824, 2014.

[23] The OpenSSL Project, 2016. (`http://www.openssl.org`)

[24] P. J. Pinero, J. A. Cortes, J. Malgosa, F. J. Canete, P. Manzanares, and L. Diez, "Analysis and improvement of multicast communications in HomePlug AV-based in-home networks," *Computer Networks*, vol. 62, pp. 89–100, 2014.

[25] A. Pinto and J. W. Atwood, "Secure multicast in IPTV services," *Computer Networks*, vol. 54, no. 10, pp. 1531–1542, 2010.

[26] Y. Rengasamy and A. Magrica, "Priority based resource allocation in hybrid network environment," *International Journal of Electronics and Information Engineering*, vol. 3, no. 2, pp. 81–90, 2015.

[27] A. Shamir, "How to share a secret," *Communications of the ACM*, pp. 612–613, 1976.

[28] A. Shimizu and S. Miyaguchi, "Fast data encipherment algorithm FEAL," in *Advances in Cryptology (EUROCRYPT'87)*, pp. 267–278, Springer, 1987.

[29] C. Y. Sun and C. C. Chang, "Cryptanalysis of a secure and efficient authentication scheme for access control in mobile pay-TV systems," *International Journal of Network Security*, vol. 18, no. 3, pp. 594–596, 2016.

[30] D. Tian, J. Zhou, Y. Wang, H. Xia, Z. Yi, and H. Liu, "Optimal epidemic broadcasting for vehicular ad hoc networks," *International Journal of Communication Systems*, vol. 27, no. 9, pp. 1220–1242, 2014.

[31] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 16–30, 2000.

Kuo-Jui Wei received the MS degree in information engineering and computer science in 2011. He is currently pursuing her Ph.D. degree in Information Engineering and Computer Science in Feng Chia University, Taichung, Taiwan. His current research interests include information security and mobile communications.

Jung-San Lee received his Ph.D. degree in computer science and information engineering from National Chung Cheng University, Taiwan in 2008. Since 2012, he has worked as an associate professor in the Department of Information Engineering and Computer Science at Feng Chia University, Taichung, Taiwan. His current research interests include image processing, information security, and mobile communications.

Bo Li received the Bachelor degree of information security from Tongji University, China in 2011. She is currently pursuing her Ph.D. degree in Electrical Engineering and Computer Science in Vanderbilt University, Tennessee, USA. Her current research interests include secret sharing technique and mobile communications.

High-speed Firewall Rule Verification with $O(1)$ Worst-case Access Time

Suchart Khummanee and Kitt Tientanopajai
(Corresponding author: Suchart Khummanee)

Department of Computer Engineering, Khon Kaen University
Khon Kaen 40002, Thailand
(Email: khummanee@gmail.com)

(Received Nov. 6, 2015; revised and accepted Jan. 23, 2016)

Abstract

Firewalls enforced by rules are a security measure for verifying huge packets at gateway networks. Therefore, they probably act as bottlenecks of the networks. In this paper, we have presented several techniques to improve the speed of firewall rule verification with *O(1)* worst-case access time. The techniques are: policy mapping (PMAP), sparse matrix packing firewall (SMPF), perfect hashing firewall (PHF) and minimal perfect hashing firewall (MPHF). The experimental results show that they are as fast as IPSet, one of the most famous high-speed firewalls at present. However, they can get rid of IPSet limitations such as IP address classes, subnet size of each rule set and so on. Besides, on average, SMPF, MPHF and PFH can reduce the amount of memory usage of PMAP by 99.9, 87.7 and 62.3 percent respectively.

Keywords: Minimal perfect hashing firewall, perfect hashing firewall, policy mapping, rule verification, sparse matrix packing firewall

1 Introduction

The amount of traffic currently flowing in and out over the networks is massive (Gigabit per second: Gbps). Firewalls equipped at the network gateways also need to process increasingly high-volumes of data. This may lead to bottlenecks on the networks, because firewalls enforced by rules must verify contents in the header of every packet. In traditional firewalls, the packet is verified against firewall rules from the top to bottom as r_1, r_2, $\cdots$, r_{n-1} by order. A summation of packet sending and receiving over networks is proportional to the ability of firewall rule verification. For example, if a network infrastructure is capable to provide throughput over 1 Gbps, but the firewall can verify packets of about 0.5 Gbps only, the rest of packets (50%) being processed will be delayed and collected for processing in the next second. While the packets are increasing steadily, and have not been processed, the connections will be reset by the network protocol (Connection timeout), and connections are lost automatically. It results in a waste of time without benefits, because of applications have to retransmit new connections again and again. In addition, the number of firewall rules is also critical to the overall performance of firewalls. In a large company, for example, there are about 2,000 firewall rules. Each rule is checked 6 times per one packet; therefore, the firewall rules must be matched the total of 12,000 times per one packet. In traditional firewalls as Netfilter/IPTables [29], it verifies rules by the order, thus the worst-case access time is $O(n)$, where n is the number of firewall rules.

To modify traditional firewall rule verification, Alex X. Liu et al. [21] proposed a new concept called the firewall rule decision state diagram (FDD) instead of verification by the sequence. It applied to several contributions such as the verification of distributed firewalls [12], firewall policy queries [23], diverse firewall design [22] and so on [20]. Although, FDD clearly shows the firewall rule conflict paths, the computational complexity of rule verification is $O(n^d)$, where n is the number of rules and d is the number of checked fields in each firewall rule. After, Acharya and Gouda [1] proposed a linear time algorithm that reduced the time complexity of FDD from as $O(n^d)$ to $O(nd)$. Next, Hamed et al. [13] contributed an algorithm for optimizing unwanted rejection flows with a dynamic packet filtering by statistical search and trees structures; moreover, they also improved the speed of the computational complexity of packet matching as $O(n\,log\,n)$.

The speed of rule verification has been improving continuously. Rovniagin and Wool [28] reduced the searching time of rule matching for $O(log\,n)$ and consumed a suitable memory for $O(n^4)$ by the Geometric Efficient Matching (GEM) technique. Khummanee et al. [18] presented the single domain decision approach (SDD) to eliminate firewall rule conflicts and the time for searching is $O(log\,n)$ by using the tree structure. The Tree-Rule firewall running on a cloud computing was introduced by Xiangjian et al. [14]. They showed that Tree-Rule used tree structure had absolutely no conflicts or redundant rules, and they evaluated their speed of verification as $O(log\, n)$. According to the improved verification speed against tree structures, HiPAC [3, 15] issued the sophisticated HiPAC packet classification algorithm and the advanced tree structure. The result showed that HiPAC could improve the speed of rule matching from as $O(log\, n)$ to $O(log\, w)$, where w is the bit width of the packet field. Currently, the state of the art of high-speed firewall rule verification is IPSet [24] which is the top of high-speed firewall open source. It applied the perfect hashing function [10] for matching the firewall rules, their experimental result is $O(1)$. However, it has few drawbacks. First, the rules must be grouped to be a set of rules before deploying them to the perfect hashing function. Second, it is available for the IP class C and B only, excluding A. If anyone would like to use an IP class A, it needs to divide the IP class A to an IP class C first. Finally, designing rule of IPSet is not easy to understand, it needs an expert in firewall rule relationships. According to the limitations of the IPSet, Khummanee [19] proposed the policy mapping algorithm (PMAP) to solve the drawbacks of IPSet. It is also $O(1)$ of the matching time like IPSet; however, it has still a problem about the memory usage. We conclude the development on the speed of firewall rule verification in Table 1.

In this paper, we have optimized the space complexity of PMAP and compare it against other techniques that are $O(1)$ worst-case access time on the firewall rule verification. The rest of the paper is organized as follows: Section 2 presents the related work, high-speed firewall designs are explained in Section 3. In Section 4, we demonstrate the performance evaluation. Finally, we give conclusions and future work in Section 5.

Table 1: History of the speed of rule verification

No.	Article's Name	Time
1	FDD[22]	$O(n^d)$
2	Linear-Time[1]	$O(n * d)$
3	IPTables[29]	$O(n)$
4	Dynamic Opt[13]	$O(n\, log\, n)$
5	GEM[28], SDD[18], Tree-Rule[14]	$O(log\, n)$
6	HiPAC[3]	$O(log\, w)$
7	IPSet[24], PMAP[19]	$O(1)$

Sort by lowest (No. 1) to highest speed

2 Related Work

2.1 Firewall Basic

Basically, a firewall rule consists of six parts: Source IP address (SIP), Destination IP address (DIP), Source Port (SP), Destination Port (DP), Protocol (Pro) and Action (Act). The first five parts are called the predicate, and the last part is called the action. Every packet flowing in the networks is matched against SIP, DIP, SP, DP and Pro of a firewall rule (r_n, $n \in \mathbb{Z}^+$) by order. If a packet matches all parts of the predicate ($\forall p_{x_i} \in r_{n_i}$, x and $n \in \mathbb{Z}^+$, $i \in \{SIP, DIP, \cdots, Act\}$), an action is operated by an Act part (accept or deny). A packet evaluated to be acceptable (accept: a) is forwarded to a destination IP address defined in its header field (DIP). On the other hand, an unacceptable packet (discard or deny: d) is automatically dropped [19]. According to Table 2, rule No. 1 (r_1) represents that the firewall allows source IP addresses ranging from 0.0.0.10 to 0.0.0.30 (21 hosts) onto destination IP addresses in the range between 0.0.0.20 and 0.0.0.30 (11 hosts), any source ports ($* \in \{0, 1, \cdots, 65, 535$ $\}$), a destination port number 80, and TCP or UDP protocol can pass through the firewall ($Act \rightarrow a$). In contrast, rule No. 3 (r_3) drops every packet from source IP addresses ranging from 0.0.0.1 to 0.0.0.40 onto destination IP addresses in the range of 0.0.0.25 to 0.0.0.35, any source port (*), a destination port number 80, and both protocols. Moreover, firewalls have an option that allows an administrator to set the final rule. This rule is to drop all packets that are not explicitly allowed ($r_1 - r_{n-1}$) at the bottom of the rule list (r_n) [31].

Table 2: Firewall rule examples

No.	***SIP***	***DIP***	***SP***	***DP***	***Pro***	***Act***
r_1	0.0.0.10-30	0.0.0.20-30	*	80	*	a
r_2	0.0.0.1-15	0.0.0.50-60	*	25-30	*	a
r_3	0.0.0.1-40	0.0.0.25-35	*	80	*	d
r_4	0.0.0.15-45	0.0.0.1-100	*	60-90	*	d
r_{n-1}	...	...	...	...	...	...
r_n	*	*	*	*	*	d

$*(SIP, DIP) = 0$ - $2^{32} - 1$, $*(SP, DP) = 0$ - $2^{16} - 1$, $*(Pro) =$ TCP and UDP, $a =$ accept, $d =$ deny

2.2 Minimal and Perfect Hashing

The firewall problem adapted with tree structures is to speed up of searching, because of the trees have a limit of the worse-case runtime as $O(log\, n)$ only. In this section, we represent the data structures that can be searched in $O(1)$ time, this concept is referred to as hashing. A ***hash table*** is a collection of stored data items. Let $\mathbf{U}$ be universal keys $= \{0, 1, \cdots, m - 1\}$, where $m \in \mathbb{N}_0$. $\mathbf{T}$ denotes a hash table $T[0, 1, \cdots, m-1]$, in which each position, or slot, corresponds to a key k_i, where $i \in \mathbb{N}_1$, in the universal U. Initially, the hash table contains no values, thus every slot is empty (NULL) as shown in Figure 1. Mapping between an value and a slot where that value belongs in the hash table is called the ***hash function (h)***. The hash function (h) computes the slot from the key k_i. In other words, h maps the universe U of keys into the slot of hash table $T[0, 1, \cdots, m - 1]$, denotes as $h : U \rightarrow \{0, 1, \cdots, m - 1\}$, where the $|m|$ of the hash table T is commonly more less than $|U|$. The function (h) hashes an value with the key k_i to slot $h(k_i)$ of T, we say that

$h(k_i)$ is the ***hash value*** of the key k_i. Figure 1 illustrates the basic concept of hashing. The hash function can reduce the size of $|U|$ to size m.

The first simply hash function, sometimes referred to as the "remainder method", divides an key k_i by the table size ($|T|$), returning remainder as its hash value $\boldsymbol{h}(k_i) = (k_i \ \% \ |T|)$. Assume that we have the set of keys and values (k_i:'$value_i$') of the ASCII code = {65:'A', 72:'H', 71:'G', 74:'J', 85:'U'}, and $|T| = 10$. Thus, the results of the hash values for our example: 5 (65 % 10), 2, 1, 4 and 5 respectively. Note that 5 of the 10 slots are now occupied. This is referred to as the ***load factor***, and is normally denoted by $\lambda = \frac{number\_of\_values}{table\_size}$. For this example, $\lambda = \frac{5}{10} = 0.5$. Once the hash values have been executed, we can insert each value into the hash table at the assigned position as shown in Figure 2. This h works well when each value is mapped to a unique position in T. However, in this example, there are two keys hashed to the same slot ($h(k_1) = h(k_5) = 5$). We refer this situation to as a ***collision*** (also called a "clash"). We need to select a systematic approach for replacing the second value ($h(k_5) = h(85)$) in the hash table by without overlapping with the others. Fortunately, effective techniques are unfolded in several data structures and algorithm books [7, 30]. In this section, we present the simplest technique to resolve the collision, called 'chaining' only.

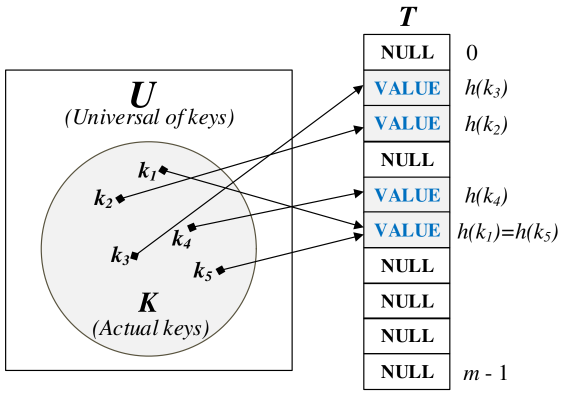


Figure 1: Using hash function (h) maps actual keys with values to hash-table slots

In the ***chaining*** approach, we put all values that hash to the same slot in T into the chain of the link list as Figure 2. The slot $h(k_i) = 5$ contains a pointer to the head of the link list where all values hashed by $h(k_i) = 5$ are stored, the tail of the link list always contains NULL. Unfortunately, the worst-case searching time is proportional to the length of the link list, that is $O(n)$, where $n <= k$, $k = |K|$. There are several effective techniques that solve the collision, such as open addressing, liner probing, and so on [7, 30]; however, these topics are beyond this paper.

The remainder method is a single hash function or fixed hash function. To yield an average retrieval time where the fixed hash function is $\Theta(n)$ in which all keys are hashed to the same slot, means a high collision rate.

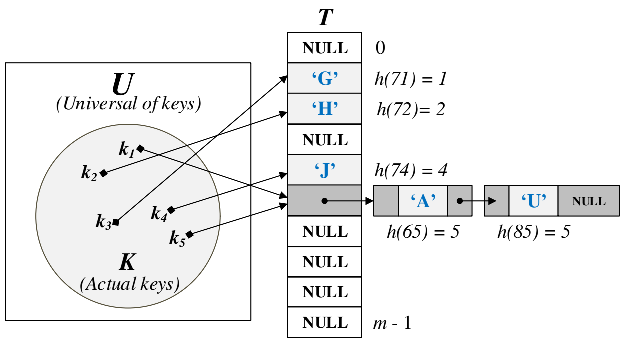


Figure 2: Five hash values $h(k_1), \cdots, h(k_5)$ in hash table

The effective technique used to improve the vulnerable fixed hash is to choose a hash function at random from a family of hash functions instead of the fixed hash function. This technique, called ***universal hashing***, has an affect on a good average access time, and guarantees a low number of collisions.

Let U be the set of universe keys and $\mathcal{H}$ be a finite collection of hashing functions mapping U into the range of integer $M = \{0, 1, \cdots, m-1\}$. Then $\mathcal{H}$ is called a universal family if $\forall x, y \in U, x \neq y : |\{h \in \mathcal{H} : h(x) = h(y)\}| = \dfrac{|\mathcal{H}|}{m}$. In the other words, the probability of a collision for any two different keys x and y hashed by a hashing function randomly which is chosen from $\mathcal{H} = \dfrac{1}{m}$. Choosing the randomness hashing function family and the uniform hashing are shown in [7]. We can see that the universal hashing is the best situation for the $average-case$ performance; however, the universal hashing can also improve $worst-case$ performance when the set of keys is $static$ (i.e. the set of keys are known). It is possible to compute the effective hashing function that can find any key in one probe ($O(1)$) in the hash table and guarantees that it has no collisions at all. Such hash functions are called ***perfect hashing***.

Let S be a set of keys, we say that a hash function h: $U \to M = \{0, 1, \cdots, m-1\}$ is a perfect hash function for S if h is injection on S, that is, there are no collisions among the keys in S if $\forall x, y \in S, x \neq y, h(x) \neq h(y)$ [10]. The Figure 3 (a) illustrates a perfect hash function concept with no collision. Any $k_i \in S$ can be retrieved from the hash table T by hash function h only once (single probe). If $n = (m-1)$, then the hash table size ($|T|$) is equal to the key size ($|S|$). We say that h is ***minimal perfect hash function*** of S as shown in Figure 3 (b). The minimal perfect hash functions are designed to totally avoid the wasted space and time problem. They are also widely applied for several applications where keys are static sets, such as words in natural languages, reserved words in programming languages, data mining and also network security.

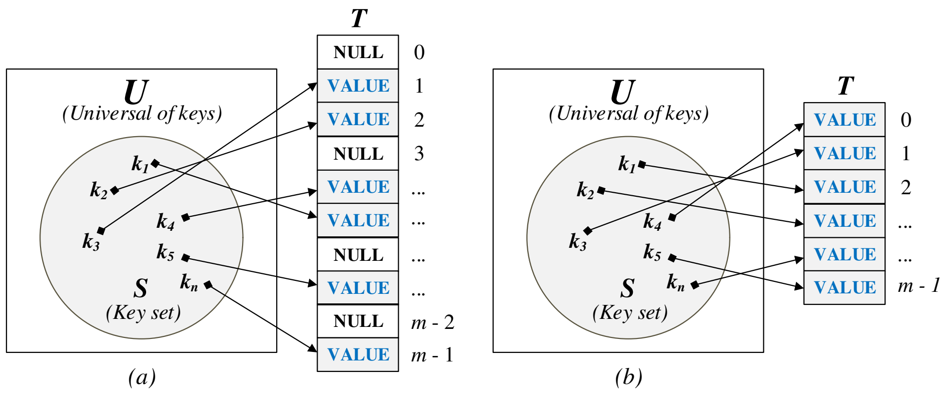


Figure 3: (a) Perfect hash function, and (b) Minimal perfect hash function

2.3 Sparse Matrix and Storage Formats

A ***sparse matrix*** is a matrix where the majority of elements have zero. In other words, the matrix which has a very few nonzero elements. Let A be a matrix and $A \in \mathbb{R}^{m\,x\,n}$, where $m, n \in \mathbb{N}_0$. We say that A is sparse if its number of nonzero entries is $O(min\{m, n\})$ [8, 26]. The following is example of a sparse matrix as shown in Figure 4.

$$\begin{array}{cl} & \quad\begin{array}{ccccc}(0,0) & & n & & \end{array} \\ \boldsymbol{A} = m & \begin{bmatrix} 11 & 0 & 0 & 0 & 0 \\ 0 & 22 & 23 & 0 & 0 \\ 0 & 32 & 33 & 34 & 0 \\ 0 & 0 & 0 & 44 & 0 \\ 0 & 0 & 0 & 0 & 55 \end{bmatrix} \in \mathbf{R}^{m=5,\,n=5} \\ & \qquad\qquad\qquad\qquad (4,4) \end{array}$$

Figure 4: A sparse matrix (m = rows, n = columns) contains only 8 nonzero elements.

The sparse matrices are involved a large number of applications, especially in science and engineering. Basically, the sparse matrices are represented in the two-dimensional array. Thus, the capacity of memory allocated for a matrix is m x n x s, where s is the size of the data type (bytes) required to store the value. In the following sparse matrix A above, there are 5 rows (m) and 5 columns (n). We need to store the integer values, and then the memory consumption of A is 5 x 5 x 2 = 50 bytes; therefore, the space complexity is $O(m$ x $n)$. Representing the sparse matrices in array structures, each element in the array is presented by $A_{i,j}$ to access an address stored the data of the matrices. In general, i indicates to the row index, and j means the column index. To perform any operations on a sparse matrix, such as multiplying the elements, the time complexity will be $O(n^2)$, where $n = m$, because the operations that are executed on matrices need to operate in two nested loops. However, provided that the operation has known the index of i and j to access an element in the matrix, the worst-case access time will be $O(1)$.

Sparse matrix storage formats. Reduction of space and time complexity of a sparse matrix can be realized by collecting only the nonzero elements. The data structures for supporting this approach will be more complex to access the individual elements, and will be able to be restored to the original matrix properly [27, 33]. There are two groups of storage formats for storing sparse matrices: the first group is designed for efficiently modification, such as DOK (Dictionary of keys), LIL (List of lists), COO (Coordinate list) and so forth, and the second group is for access and matrix operations, such as CSR (Compressed Sparse Row) or CSC (Compressed Sparse Column). In this paper, we have showed the CSR format only, because of CSR is the most general format and widely used for implementing and referring. CSR places a set of nonzero (nz) of the matrix rows to contiguous memory spaces (array) which have the memory size to be $|nz|$. Suppose that we have a nonsymmetric sparse matrix $A \in \mathbb{R}^{m=5, x\, n=5}$ in Figure 4, and we then create three arrays for storing its: one of floating-point numbers (val), and two for integers (col_ind and row_ptr) as shown in Figure 5. The total size the arrays are: $val = |nz| = 8$, $col\_ind = |nz| = 8$ and $row\_ptr = m + 1 = 6$ respectively. The val maintains only nonzero elements of matrix A by collecting in row order. The col_ind array stores the column indexes of the elements in the val array, that is, if $val(k) = A_{i,j}$, then $col\_ind(k) = j$ as well. Last, row_ptr keeps the starting location of each row stored in val, that is, if $val(k) = A_{i,j}$, then $row\_prt(i) \leq k < row\_prt(i + 1)$. This storage format can save the space capacity to store the sparse matrix A from n^2 to 2 x $|nz| + m + 1$. In this example, it can be reduced from 50 bytes (5 x 5 x 2 bytes) to 44 bytes ((2 x 8 + 5 + 1) x 2 bytes). The time complexity of CSR to operate any operations such as the matrix-vector multiplication is $O(n$ x $m)$, where

$n = |val|$, $m = |row\_ptr|$.

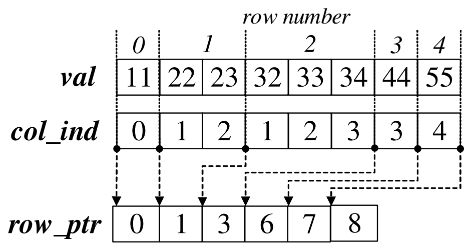


Figure 5: The CSR format for matrix A is specified by the arrays val, col_ind and row_ptr

Sparse matrix compression. To improve the memory space of the sparse matrix storage formats, Horowitz et al. [16] showed a simple and effective sparse matrix compression, called the nonzero-term method (NM). NM stores nonzero elements of a sparse matrix into a (row_index, column_index, value) format, for example, (1,2,4) means the row = 1, column = 2 and nonzero value = 4. Although, NM is easily for implementation, it takes the linear searching time against the nonzero elements. Ziegler's method (ZM) [32] is widely famous technique that applies shift-left and merging instructions to compress the nonzero values. In addition, rehash method (RM) [6] is to accomplish the sparse binary-matrix compression. This method is only applied with the binary number. Ji-Han Jiang et al. [17] extended the rehash method of RM by using the random hash function. Aiyoub Farzaneh et al. [9] contributed the Compressed Sparse Vector (CSV), which reduces the storage value of large non-symmetric sparse matrices more than CSR.

3 High-speed Firewall Structures

In this section, we depict the design of high-speed firewall structures. The high-speed firewalls in this paper mean firewalls that can match a packet against the predefined firewall rule with $O(1)$ worst-case access time. We represent four high-speed firewall structures: the policy mapping firewall (PMAP) [19], sparse matrix packing firewall (SMPF), perfect hashing firewall (PHF) and minimal perfect hashing firewall (MPHF) successively.

3.1 Key Contributions

We make four major contributions as follows:

1) Firstly, we optimize the memory space of the policy mapping (PMAP) proposed in [19] by applying the sparse matrix compression approaches, called SMPF;

2) Propose new techniques that are adapted from perfect hashing and minimal perfect hashing to improve the high-speed firewall rule verification, called PHF and MPHF;

3) Evaluate and compare the performance of all proposed techniques including PMAP, SMPF, PHF, MPHF, and also IPSet;

4) And finally, conclude the comparison results in several aspects.

3.2 High-speed Firewall Designing

There are six milestones to design high-speed firewalls:

Step 1. Designs a firewall rule user interface, in this step we choose the Rule-Base firewall (traditional style), because of it is popularly used nowadays as shown in Figure 6 in the Step 1;

Step 2. Builds a decision state diagram structure (DSD) from the rule list in the Step 1 by using the firewall decision state diagram algorithm (FDSD);

Step 3. Maps the DSD from the Step 2 to the array structures by the policy mapping algorithm (PMAP);

Step 3.1. Packs the array structures from the Step 3 to SMPF by the sparse matrix compression technique;

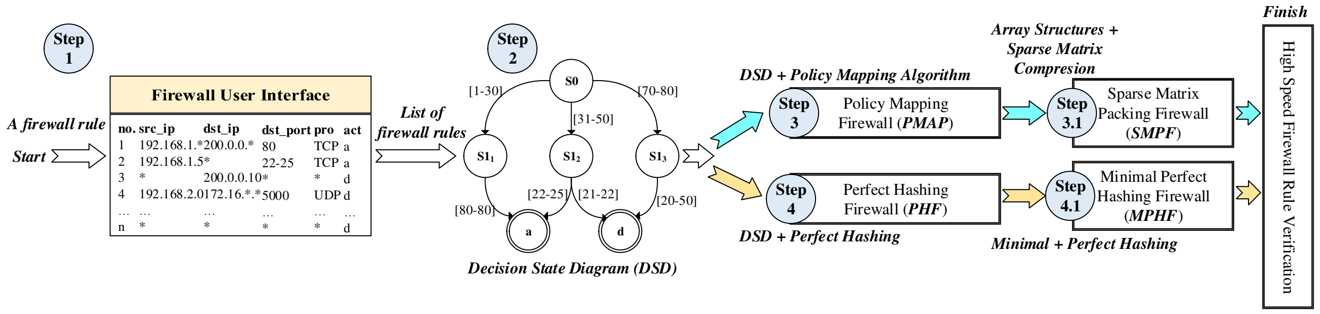
Step 4. Creates keys and values from DSD to build the perfect hashing firewall (PHF);

Step 4,1. Compacts the perfect hashing tables from the Step 4 to the minimal perfect hashing firewall (MPHF).

According to the steps of high-speed firewall design, we thoroughly describe each step like this:

Step 1: Designing the firewall user interface.
Nowadays, almost all firewall user interfaces are Rule-Base or Rule-List. The interfaces are displayed in a tuple format compounded from SIP, DIP, SP, DP, Pro and Act as {192.168.1.0-255, *.*.*.*, 1234, 80, *, a}. They have been influenced by the nature of reading and writing from left-to-right and top-to-bottom. Other user interface aspects, several researchers tried to suggest new firewall interfaces like [23] or [14]; however, they were not popular. Therefore, we still use the Rule-Base interface to make firewall rules in the Step 1. In order to easily describe firewall structures, we have presented easy firewall rules that consist of five fields: SIP, DIP, DP, Pro and Act as shown in Table 3. We also transform both SIP and DIP from the IPv4 addressing format to the positive decimal format by the equation to as $octate_4$ x 2^{24} + $octate_3$ x 2^{16} + $octate_2$ x 2^8 + $octate_1$ x 2^0. An IP address 0.0.0.10, for example, is transformed to 0_4 x 2^{24} + 0_3 x 2^{16} + 0_2 x 2^8 + 10_1 x 2^0 = 10.

Step 2: Building the DSD.
The decision state diagram (DSD) is built from the firewall decision state diagram algorithm (FDSD)


Step 1
A firewall rule
Start
Firewall User Interface
Step 2
S0
[1-30]
[31-50]
[70-80]
S1₁
S1₂
S1₃
[80-80]
[22-25]
[21-22]
[20-50]
a
d
List of firewall rules
Decision State Diagram (DSD)
DSD + Policy Mapping Algorithm
Step 3
Policy Mapping Firewall (PMAP)
Step 4
Perfect Hashing Firewall (PHF)
DSD + Perfect Hashing
Array Structures + Sparse Matrix Compresion
Step 3.1
Sparse Matrix Packing Firewall (SMPF)
Step 4.1
Minimal Perfect Hashing Firewall (MPHF)
Minimal + Perfect Hashing
Finish
High Speed Firewall Rule Verification

no.	src_ip	dst_ip	dst_port	pro	act
1	192.168.1.*	200.0.0.*	80	TCP	a
2	192.168.1.5	*	22-25	TCP	a
3	*	200.0.0.10	*	*	d
4	192.168.2.0	172.16.*.*	5000	UDP	d
…	…	…	…	…	…
n	*	*	*	*	d

Figure 6: Steps of high-speed firewall designing

Table 3: Easy firewall rules for building DSD

No.	*SIP*	*DIP*	*DP*	*Pro*	*Act*
r_1	10 - 30	20 - 30	80	*	***a***
r_2	1 - 15	50 - 60	25 - 30	*	***a***
r_3	1 - 40	25 - 35	80	*	***d***
r_4	15 - 45	1 - 100	60 - 90	*	***d***

proposed by Khummanee et al. [19]. It has several roles, for example, it makes sure that the firewall decision paths are not any confusion, and eliminates decision paths duplicated in firewall rules. The details on how to build the DSD is described in [19]. In this paper, we only show the final decision state diagram which was created from firewall rules in Table 3 successfully as shown in Figure 7. Referring to the DSD in Figure 7, suppose that there is a packet arriving from somewhere to our networks. It consists of DP = 25, DIP = 0.0.0.55, SIP = 0.0.0.10 and Pro = TCP. Thus, the firewall makes a decision to allow this packet into the networks because it follows the 1st state decision path of the DSD diagram (25 $\in$ {25-30}, 55 $\in$ {50-60}, 10 $\in$ {1-15} and TCP $\in$ {TCP, UDP} $\rightarrow a$). Another example, if a packet is composed of DP = 80, DIP = 0.0.0.10, SIP = 0.0.0.45 and Pro = TCP. This packet thereby will be dropped by the 3rd path of DSD.

Step 3: Mapping DSD to array structures.

In this step, we map the DSD from the Step 2 to arrays. The DSD has four levels are DP, DIP, SIP and Pro respectively. The DP level (the 1st level) in Figure 7 is mapped to one dimension array, named S0-S1 as shown in Figure 8. The algorithm for mapping DSD to array structures is illustrated in [19]. The memory size of S0-S1 equals to 65,536 elements. Each element has 16-bits integer; therefore, the total of memory size of the S0-S1 array is $\approx$ 131 kilobyte (KB). The 2nd level (DIP) is transformed to the three-dimensional array of two cubes namely S1-S2$_{upper}$ and S1-S2$_{lower}$. The S1-S2$_{upper}$ is used to store $octate_3$ (x-axis) and $octate_4$ (y-axis), and the S1-S2$_{lower}$ collects $octate_1$ (x-axis) and $octate_2$ (y-axis) of DIP. The z-axis of both three-dimensional arrays is referred to the decision state path (state path number) of DSD. The 3rd level (SIP) is as similar as DIP. The algorithm converts SIP level to S2-S3 arrays, that is, S2-S3$_{upper}$ maintains $octate_3$ and $octate_4$, and S2-S3$_{lower}$ collects $octate_1$ and $octate_2$. The total of maximum memory usage of both S1-S2 and S2-S3 is $\approx$ 16.97 GB (x-axis_size $\times$ y-axis_size $\times$ data_size $\times$ state_path_size = 256 $\times$ 256 $\times$ 16 $\times$ 65,536 bit) to deal with 65,536 state paths of the firewall rule. The final state level of DSD (4th) is Pro, it is mapped to two dimensional array, namely S3-S4. The total size of S3-S4 is about $\approx$ 33.55 MB. The x-axis of S3-S4 array is used to store the decision (Act) of the firewall rule state path, and y-axis points to the firewall rule state path.

For example, the 4st state path of DSD diagram in Figure 7, DP is equal to {80-80}, it means a destination port number 80. Thus, S0-S1[80] is set to be 3 (the state path in DIP level). DIP is subset of {20-30}, it indicates the range of destination IP addresses between 0.0.0.20 and 0.0.0.30. Consequently, S1-S2[0][0][3]$_{upper}$ is equal to $'X'$ (X = don't care term), S1-S2[20-30][0][3]$_{lower}$ are assigned to be 4 (state path of SIP level). Likewise, the SIP is subset of source IP addresses ranging from 0.0.0.1 to 0.0.0.9 ($SIP \in$ {1-9}). Therefore, S2-S3[0][0][4]$_{upper}$ is stored $'X'$, and S2-S3[1-9][0][4]$_{lower}$ keep the positive integers (the number 4) that point to an array stored protocols in S3-S4. Lastly, the 4th state path of Pro level is the subset of both TCP and UDP, so S3-S4[4][6] (TCP) and S3-S4[4][17] (UDP) are assigned to be $'d'$ (deny) as shown in Figure 8. The sum of all memory space used to support the number of 65,536 rules is about 17 GB.

Step 3.1: Packing array data structures.

Notice that the stored values in array data structures in Figure 8 after finishing the Step 3 are similar to the sparse matrices discussed in Section 2.3. Furthermore, the total memory size of the arrays is not optimal yet. To optimize the memory usage of arrays, we have applied the sparse matrix

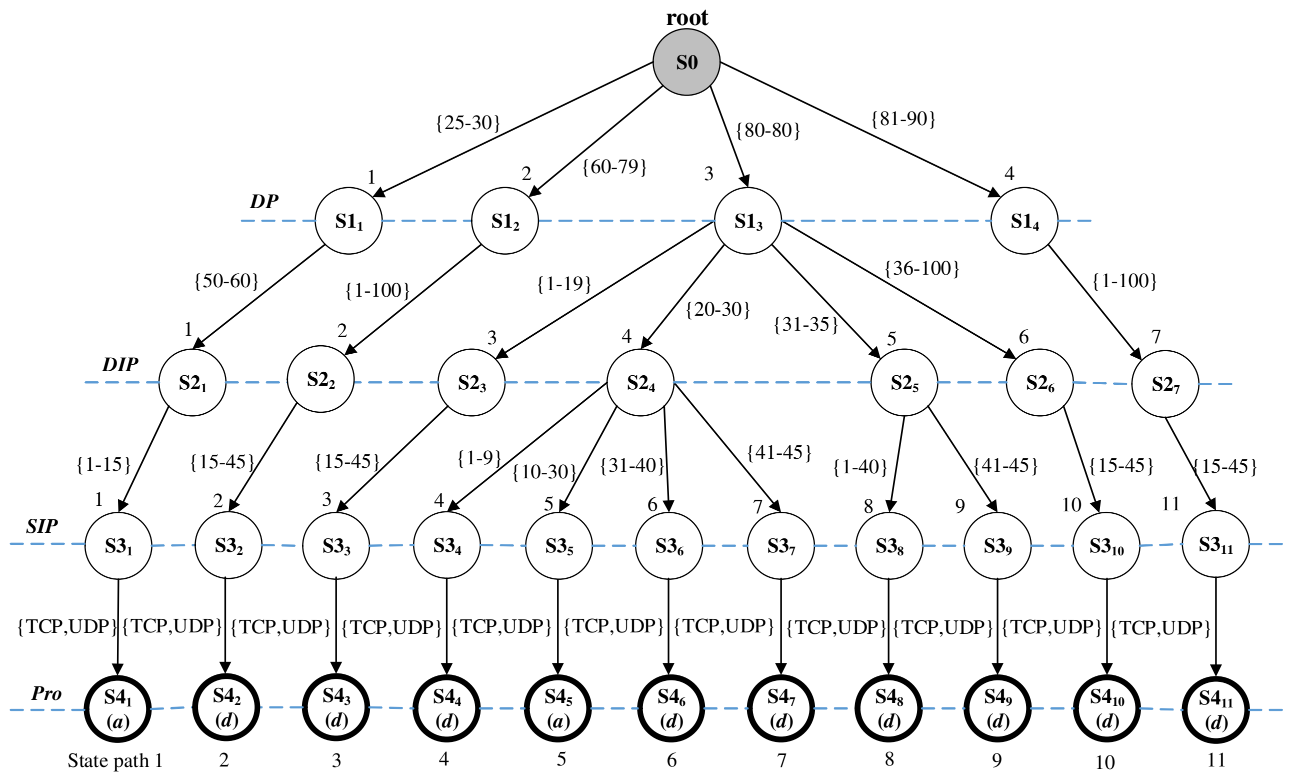


Figure 7: The final decision state diagram (DSD)

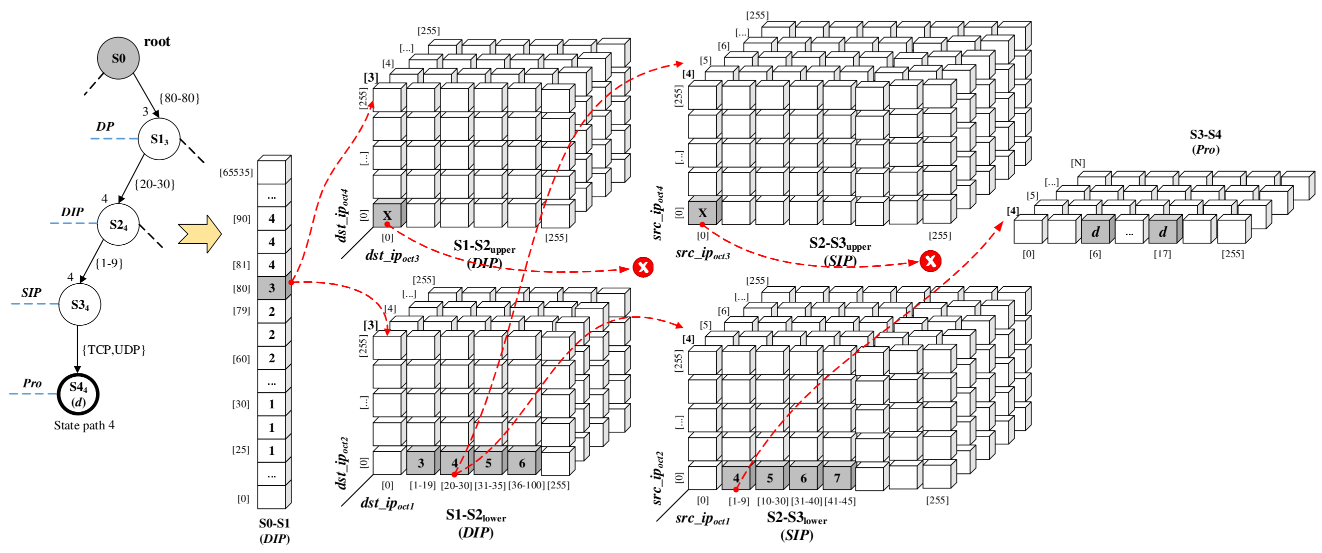


Figure 8: Mapping the 4th state path of DSD to array structures

compression techniques to reduce them (also optimize PMAP), referred to as Sparse Matrix Packing Firewall (SMPF). The concepts of the sparse matrix packing are adapted from [5, 2]; however, they have limitations for directly deploying with the firewall rule. For example, they only support two-dimensional arrays, and do not appropriate for firewall's array data structures which are the three-dimensional array. Besides, they are tested on the small sparse matrices, but sparse matrices for the firewall rules are very large. Generic sparse matrices can freely swap any positions of rows, but the rows in sparse matrices of firewall rule cannot arbitrarily change because the matching may be wrong as a result. In order for the sparse matrix to be able to perfectly handle firewall rule structures, we have improved the mechanisms to store and pack sparse matrices as shown in Algorithm 1, referred to as the Sparse Matrix Packing (SMP).

Algorithm 1 Sparse matrix packing algorithm (SMP)

```
1: Input: S1-S2$_{upper,lower}$, S2-S3$_{upper,lower}$, and S3-S4
2: Output: RLT$_{S1-S2_{upper,lower}}$, RLT$_{S2-S3_{upper,lower}}$, RLT$_{S3-S4}$ and 1-D$_{S1-S2,S2-S3,S3-S4}$
3: set size = 256, path = $N$ (size = array size, $N \in \mathbb{N}_1$)
4: create 1-D$_{S1-S2,S2-S3_{upper}}$,1-D$_{S1-S2,S2-S3_{lower}}$=[ ]
5: create RLT$_{S1-S2,S2-S3_{upper,lower}}$ = [path][size]
6: create 1-D$_{S3-S4}$ = [ ], RLT$_{S3-S4}$ = [path]
7: set pc, rc = 1 (pc=page count,rc=row count)
8: while path ≤ pc do
9:   while rc ≤ size do
10:     read data from S1-S2[rc][*][pc]$_{upper}$
11:     append data to 1-D$_{S1-S2_{upper}}$ consecutively
12:     add 1st index of data to RLT[pc][rc]$_{S1-S2_{upper}}$
13:     read data from S1-S2[rc][*][pc]$_{lower}$
14:     append data to 1-D$_{S1-S2_{lower}}$ consecutively
15:     add 1st index of data to RLT[pc][rc]$_{S1-S2_{lower}}$
16:     read data from S2-S3[rc][*][pc]$_{upper}$
17:     append data to 1-D$_{S2-S3_{upper}}$ consecutively
18:     add 1st index of data to RLT[pc][rc]$_{S2-S3_{upper}}$
19:     read data from S2-S3[rc][*][pc]$_{lower}$
20:     append data to 1-D$_{S2-S3_{lower}}$ consecutively
21:     add 1st index of data to RLT[pc][rc]$_{S2-S3_{lower}}$
22:     read data from S3-S4[rc][pc]
23:     append data to 1-D$_{S3-S4}$ consecutively
24:     add 1st index of data to RLT[pc][rc]$_{S3-S4}$
25:     rc++
26:   end while
27:   pc++
28: end while
29: End
```

The SMP starts with: (1) reading 3-D or 2-D sparse matrices from array data structures of the firewall rule as shown in Figure 8, (2) packing them to 1-D arrays, and (3) recording the indexes pointed to the packed 3-D or 2-D data in *Row Lookup Table* (*RLT*) used later to access the 1-D array during retrieval. The 1-D array stores non-zero items of the packed 3-D or 2-D array while the RLT is held at the starting position of each row stored in 1-D array. Packing begins by separating the 3-D or 2-D array into the tuples or records. This process is shown in Figure 9. In Figure 9 (a), suppose that the page number 4 (the stat path number 4 in SIP level of DSD) of the 3-D array is first packed into the 1-D array. The 1st packed record is S2-S3[0][*]$_{lower}$ (S2-S3[row][$column$], * = all columns), 2nd packed record is S2-S3[1][*]$_{lower}$, and final packed record is S2-S3[255][*]$_{lower}$ respectively. The page number 4 of S2-S3$_{lower}$ has only a single row that has non-zero items between the location 1 and 45. These non-zero items are placed into the first location of 1-D array, this process is shown in Figure 9 (b). To retrieve the non-zero items in 1-D array later, the RLT marks how they were packed by recording the integer position for each row of the original 2-D arrays into the RLT. For example, the value -1 in the 4th row ([4]) of RLT table indicates to the 1st row in the page no. 4 of S2-S3$_{lower}$ array (S2-S3[0][*][4]$_{lower}$) because that is the offset for replacing the non-zero items ranging from 1 to 45 (starting free slot in 1-D array - starting non-zero item list $\rightarrow$ 0 - 1 = -1). As a result, the starting free slot of 1-D array is shifted from the position 0 to 44 immediately. The next non-zero items allocated in the page no. 5 ([5]), there are ranging from the position 1 to 45. The packing process places them to 1-D array in the position 45 which is currently pointed to the starting free slot index, then the index is updated to the location 90 (45 + 45). The offset of the page no. 5 in 5th row in RLT is also recorded to 44 (starting free slot - starting non-item list $\rightarrow$ 45 - 1 = 44). The last example of the packing process, the list of non-zero items in page no. 6, is arrayed from 15 to 45 (30 positions). This list is allocated in the position 90 to 120 in the 1-D array, the starting free slot index is updated to 121, and the offset in RLT of this list (6th row) is set to 75 (90 - 15). On the other hand, the rows that have not non-zero items are set to the blank record in RLT table like the 1st, 2nd, 3rd row and so forth. To retrieve any non-zero items in the 1-D array, we use the formula as following:

Offset = RLT[Page No.][Row No.];

Index = Offset + Column;

Non-zero item = 1-D[Index]

For example, suppose that we would like to retrieve the non-zero item of the S2-S3$_{lower}$ array by the row = 0, column = 41 and page no. 5 in 1-D array. Thus, we compute the position of this item as Offset = RTL[5][0] = 44, Index = Offset + Column = 44 + 41 = 85, and then the non-zero item = 1-D[85]$_{S2-S3_{lower}}$ = 9 as shown in Figure 9. The SMP algorithm matches the firewall rules against any packet$_i$ shows in Algorithm 2.

Step 4: Building the perfect hashing firewall.

IPSet [24] running on IPTables [29] and Netfilter was successfully applied the perfect hash function to improve the speed of firewall rule verification. It offers

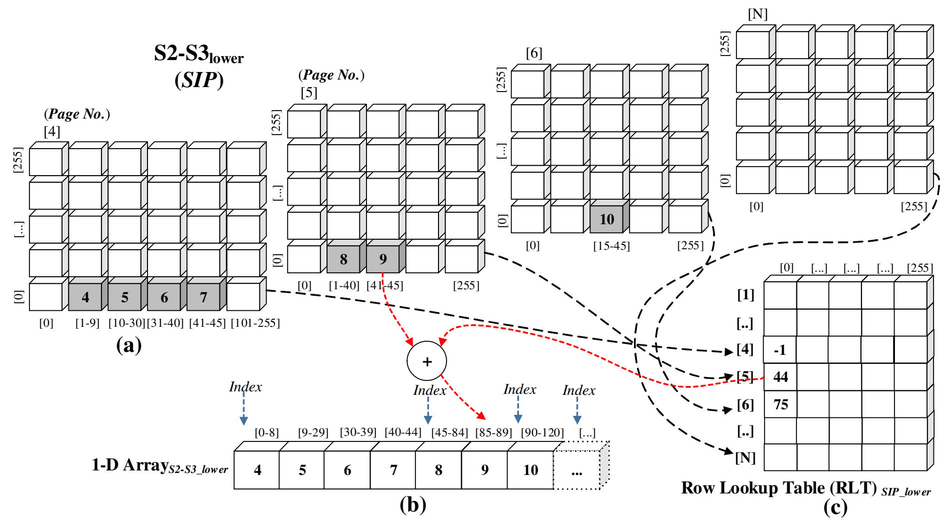


Figure 9: (a) separating the 3-D array of SIP namely S2-S3$_{lower}$ to records, (b) packing records to 1-D array, and (c) adding the starting position of each packed record to RLT

Algorithm 2 Sparse matrix matching

```
1:  Input: S0-S1, RLT$_{S1-S2_{up,lo}}$, RLT$_{S2-S3_{up,lo}}$, RLT$_{S3-S4}$ and 1-D$_{S1-S2,S2-S3,S3-S4}$, packet $p_i$
2:  Output: $a$ = accept or $d$ = deny
3:  get $DP$, $DIP$, $SIP$, $Pro \leftarrow p_i$
4:  split $DIP_{oct_1}$, $DIP_{oct_2}$, $DIP_{oct_3}$, $DIP_{oct_4} \leftarrow DIP$
5:  split $SIP_{oct_1}$, $SIP_{oct_2}$, $SIP_{oct_3}$, $SIP_{oct_4} \leftarrow SIP$
6:  page ← S0-S1[$DIP$]
7:  offset ← RTL[page][$DIP_{oct_2}$]$_{S1-S2_{lower}}$
8:  index ← offset + $DIP_{oct_1}$, p1 ← 1-D[index]$_{S1-S2_{lower}}$
9:  offset ← RTL[page][$DIP_{oct_4}$]$_{S1-S2_{upper}}$
10: index ← offset + $DIP_{oct_3}$, p2 ← 1-D[index]$_{S1-S2_{upper}}$
11: page ← get value from p1 or p2 that is not 'X'
12: offset ← RTL[page][$SIP_{oct_2}$]$_{S1-S2_{lower}}$
13: index ← offset + $SIP_{oct_1}$, p1 ← 1-D[index]$_{S2-S3_{lower}}$
14: offset ← RTL[page][$SIP_{oct_4}$]$_{S1-S2_{upper}}$
15: index ← offset + $SIP_{oct_3}$, p2 ← 1-D[index]$_{S2-S3_{upper}}$
16: page ← get value from p1 or p2 that is not 'X'
17: offset ← RTL[page]$_{S3-S4}$, index ← offset + $Pro$
18: result ← 1-D[index]$_{S3-S4}$
19: if result == 'a' then
20:     print 'accept'
21: else
22:     print 'deny' (result == 'd')
23: end if
24: End
```

several features of the keys which are used for referring to the hashing table, for instance, the key is combined by the IP address against port number (IP:Port) or IP address and port and IP address (IP:Port:IP) or etc. In this section, we have showed how to apply the perfect hash function as same as IPSet, but we have chosen the different key aspect by acquiring the key from DSD instead. With the acquisition of keys from DSD, we have picked the remarkable features which are combined to the unique key. The 3rd state path of DSD diagram in Figure 7, for example, we have jointed the port number (DP), destination IP address (DIP) and source IP address (SIP) to be the key such as $'801015'$ (DP = 80, DIP = 10, SIP = 15). Yielding to the number of keys in each state path can be computed from the number of $DPs \times DIPs \times SIPs$; therefore, the 3rd state path in Figure 7 is $1 \times 19 \times 30 = 570$ keys ($'80115'$, $'80216'$, $'80317'$, $\cdots$, $'801945'$). Notice that the number of keys which will be hashed by a perfect function (h) are huge. IPSet also faces such problem; as a result, it requires a set of rules that do not exceed over the subnet of IP class C only. To avoid this problem, we have reduced the number of keys by choosing for each state path where the action (Act) is an acceptation (a) only. For example, we have only chosen the 1st and 5th state path from all paths in Figure 7 to be the keys.

The keys got from the DSD will be hashed by the perfect hash function h for referring to any address storing a value. We have applied the hashing algorithm from [4, 11] as shown in Algorithm 3, the algorithm uses two levels of hash functions. The first function is $\boldsymbol{h}_1(0, key)$, it gets a position in an intermediate array, named G. The second function is $\boldsymbol{h}_2(d, key)$, it hashes the key and the information got from G to search the unique position for the key as shown in Figure 10. For example, both of $'801235'$ and $'256811'$ key are hashed to the same position (1234) in the intermediate table (G) by using function $\boldsymbol{h}_1$. However, the second hash function $\boldsymbol{h}_2$ hashes the keys again by combining the same position against old keys, and puts hashing results into different slots in value table

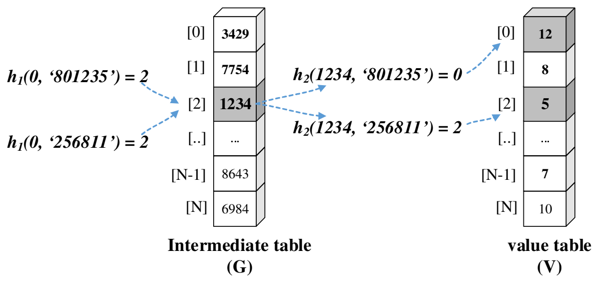


Figure 10: Perfect hash table lookup

(V) so no collision occurs. To avoid the collision from hashing, we use the FNV [25] hash function (Algorithm 4) which guarantees the very low collision rate. To look up a value in hash table G and V, we first get the DP, DIP and SIP field from a packet, and compound them to the key. For example, suppose that a packet$_i$ consists of $DP = 80$, $DIP = 200$, $SIP = 100$ and Pro = TCP, so the compounded key is $'80200100'$. We can compute a position stored the value from the key by Algorithm 5.

Algorithm 3 Perfect hash function (double hash h_1, h_2)

```
1: Input: dictionary (dict) of key and value ('key':value)
2: Output: G, V (G=intermediate, V=values table)
3: set size $\gg |dict|$
4: create bucket[size][], G[size], V[size]
5: while key$_i$ ← read dict until NULL do
6:     p ← FNV-hash(0, key$_i$) mod size
7:     bucket[p].append(key$_i$) #put all keys to buckets
8: end while
9: while key ← read bucket[$d$][*] until NULL do
10:    if $|key|$ == 1 then
11:        put 0 → G[d] ($d \in \mathbb{N}_0$)
12:        put value from dict(key) → V[d]
13:    end if
14: end while
15: while keys ← read bucket[$d$][*] until NULL do
16:    if $|keys| > 1$ then
17:        d = 1
18:        while key ← read keys until NULL do
19:            p ← FNV-hash(d, key) mod size
20:            if put d → G[p] then collision then
21:                p ← rehash(d++, key) until no collision
22:            end if
23:            V[p] ← read value from dict(key), G[p] ← d
24:        end while
25:    end if
26: end while
27: End
```

Step 4.1: Building the minimal perfect hashing firewall.
The perfect hashing is a method which guarantees no collision upon building a hash table. It is only possible to build it when we have known all of the keys in advance. Besides, the space for storing values of perfect hashing is usually more than the total size of values ($|storage| \gg |values|$) because it can reduce the percentage of collision rate. One effective method for reducing the size of memory space is the ***Minimal Perfect Hashing*** (MPH). MPH ensures that the hashing table contains one key per one slot only, and also has no free slots. In this section, we have optimized the memory space of the perfect hashing described in previous section by customizing Algorithm 3 in the line of code 3 from $set\,size \gg |dict|$ to be $set\,size = |dict|$ instead. This code determines the size of the memory to be fitted with the number of keys. The rest of the codes are as same as Algorithm 3.

Algorithm 4 The FNV algorithm

```
1: Input: d, key
2: Output: d
3: if Input has both d and key then
4:     while c ← read a char from key until NULL do
5:         d ← ((d * 0x01000193) ^ ord(c)) & 0xffffffff
6:     end while
7: else
8:     d ← 0x01000193
9: end if
10: End
```

Algorithm 5 Look up the value in hash table G, V

```
1: Input: G, V and a key
2: Output: a value
3: d = G[FNV-hash(0, key) mod len(G)]
4: if d == 0 then
5:     return V[FNV-hash(0, key) mod len(G)]
6: else
7:     if d > 0 then
8:         return V[FNV-hash(d, key) mod len(G)]
9:     else
10:        print "Not found"
11:    end if
12: end if
13: End
```

3.3 High-speed Firewall Implementation

In this section, we detail about implementing high-speed firewalls. The tested environments of this paper are similar to [19], but they are slightly different as follows:

Hardware and Software Development Tools.
The high-speed firewalls are developed on the Intel 64-bits processor, Core i7, 2GHz, installed memory (RAM) 8 GB. For the developing software, we chose Python language (version 3.4 for 64 bits), Numpy and Psutil to implement firewalls running on MS Windows 8 operating system.

Firewall Rule and Packet Generator.
In each test case, the firewall rule generator generates the randomness rules from 1,000, 2,000, 3,000, 4,000, 5,000 and 10,000, and the random packets for 10,000 packets per round by the packet generator software. We have experimented about 30 times per each algorithm, and calculated the average of speed and space for each algorithm.

4 The Performance Evaluation

All proposed firewalls are high-speed. Therefore, they can match any rules against packet$_i$ by very low constant time; in other words, they take $O(1)$ worst-case access time. However, their memory and time consumption for constructing rule structures are not equal depending on the complexity of firewall rule data structures. Thus, this paper we aim to compare the amount of memory space used for storing rule structures and the processing time for building them. In case of the computation time, there are three kinds: the time for building the decision state diagram (DSD), constructing rule structures, and matching firewall rule as shown in Table 4. The space complexity and percentage of memory space optimized are shown in Table 5.

The speed of firewall rule verification of all techniques is similar, for example, in Table 4, the average of time verifying of all approaches is quite stable at about 0.031 seconds on average. Notice that while processing the Path No. 10,000, the PMAP consumes the verifying time more than another algorithms by about 0.124 seconds because the main memory is not enough (running memory > available memory) for supporting PMAP running. As a result, the Python interpreter needs to allocate an extra memory from the virtual memory (usually the hard disk drive) to accomplish PMAP process. According to the constructing time, MPHF spends the most time on the firewall rule construction ≈ 3,983 seconds at the number of state paths (Path No.) = 5,000. Moreover, it cannot be executed successfully at the Path No. 10,000 because it takes too much constructing time, and Python interpreter sometimes crashes. The reason that it is slow is repeated searching of a free slot in a limited hash table to place the value. In contrast, PMAP is the fastest technique to build the rule data structures; on the other hand, it also consumes the most of memory usage. SMPF and PHF take the average amount of rule constructing time in the same trend by the linear manner.

In case of the memory optimization, SMPF is the best high-speed firewall to consume the minimal space stored structural rules, it can highly reduce the percentage of memory usage of PMAP by around 99.9 percent on average. While MPHF is able to optimize the memory of PMAP by about 87.7%, and PHF by about 62.3% respectively, as shown in the column of packing results of Table 5. The Keys column in Table 5 is presented the number of hashed keys by the perfect hash function of PHF and MPHF. To process the number of the state paths at 10,000, PHF and MPHF must hash the number of keys ≈ 83.4 million.

5 Conclusions and Future Work

In this paper, we have proposed several techniques to improve the high-speed firewalls that can access the data in $O(1)$ worst-case access time, namely PMAP, SMPF, PHF and MPHF. They are as fast as IPSet [24] but different in the limitations as following:

Limitations: *IPSet* needs to be set up a group of rules in the network class C only before running, and each group is not bigger than 65,536 rules per a set. It does not support the IP network class A, but we can partition them to subnets before deploying to IPSet. It is not easy to understand, the administrator must have a lot of skills about the set of rules. *PMAP* consumes a lot of memory to build the rule data structures, it only supports a maximum number of 65,536 rules. *SMPF* supports the same number of firewall rules like PMAP, and the firewall structure is quite complicated. *PHF* encounters a lot of keys effecting the firewall's performance, it should solve this problem like IPSet. *MPHF* has drawbacks like PHF, but the big problem is the time to construct firewall rules data structures.

We have concluded the overall performance and limitations of high-speed firewalls in Table 6.

Table 6: The overall performance of high-speed firewalls

Name	Space	Time complexity		Structural
		Verify	Construct	
PMAP	Fair	O(1)	Fastest	More
SMPF	Best	O(1)	Fast	Most
PHF	Good	O(1)	Faster	Much
MPHF	Better	O(1)	Slow	Much
IPSET	Good	O(1)	Faster	Much

Space = Space complexity, Verify = Verification time, Construct = Construction time, and Structural = Structural complexity

References

[1] H. B. Acharya and M. G. Gouda, "Linear-time verification of firewalls," in *International Conference on Network Protocols (ICNP'09)*, pp. 133–140, Princeton, NJ, Oct 2009.

[2] M. Aspns, A. Signell, and J. Westerholm, *Efficient Assembly of Sparse Matrices Using Hashing.* Berlin: Springer Berlin Heidelberg, 2007.

[3] M. Bellion, "High performance packet classification (HIPAC)," 2005. (`http://www.hipac.org/`)

Table 4: Time complexity results of high-speed firewals

Path No.	**Constructing time (sec)**					**Verifying time (sec)**			
	DSD	**PMAP**	**SMPF**	**PHF**	**MPHF**	**PMAP**	**SMPF**	**PHF**	**MPHF**
1,000	1.60	0.59	41.12	20.37	243.51	0.031	0.031	0.032	0.015
2,000	6.51	1.17	88.27	39.27	628.73	0.031	0.015	0.032	0.015
3,000	14.84	1.81	141.74	58.38	1477.16	0.031	0.015	0.033	0.016
4,000	26.17	2.50	204.57	77.37	1763.36	0.015	0.031	0.032	0.031
5,000	50.95	3.37	282.79	100.28	3,983.51	0.031	0.031	0.034	0.034
10,000	188.83	38.29	630.90	506.51	N/A	0.124*	0.032	0.034	N/A

Note: Path No. = the number of firewall rule state paths, DSD = decision state diagram, PMAP = policy mapping, SMPF = sparse matrix packing, PHF = perfect hashing and MPHF = minimal perfect hashing firewall, N/A = consumes too much processing time, * = main + virtual memory access time

Table 5: Space complexity and optimized memory results of high-speed firewalls

Path No.	**Constructing space (MB)**				**Packing result (%)**			**Keys (million)**
	PMAP	**SMPF**	**PHF**	**MPHF**	**SMPF**	**PHF**	**MPHF**	**PHF&MPHF**
1,000	1,043.38	0.58	402.65	124.20	99.94	61.40	88.09	7,762,577
2,000	2,085.67	0.65	805.30	255.92	99.96	61.38	87.72	15,995,188
3,000	3,113.80	0.72	1,207.95	365.99	99.97	61.20	88.24	22,874,422
4,000	4,135.63	0.79	1,610.61	534.10	99.98	61.05	87.08	33,381,606
5,000	5,163.76	0.86	2,013.26	634.31	99.98	61.01	87.71	39,644,683
10,000	10,095.22	1.18	3,221.22	N/A	99.98	68.09	N/A	83,447,420

Note: Packing result = percentage of firewall rule compression from PMAP, Keys = the number of hashed keys, N/A = cannot build rule structures resulting from Table 4.

[4] F. C. Botelho and N. Ziviani, "External perfect hashing for very large key sets," in *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management (CIKM'07)*, pp. 653–662, Lisboa, Portugal, Nov 2007.

[5] M. D. Brain and A. L. Tharp, "Perfect hashing using sparse matrix packing," *Information Systems*, vol. 15, no. 3, pp. 281–290, 1990.

[6] C. C. Chang, J. H. Jiang, and T. S. Chen, *Rehash method: A compression technique of sparse binary-matrices.* National Chung Cheng University, Chiayi, Taiwan: Technical Report of the Department of Computer Science and Information Engineering, 1996.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms Third Edition.* Massachusetts Institute of Technology: The MIT Press, 2009.

[8] S. Debasis, *Classic Data Structures.* Delhi, India: PHI Learning; 2nd edition edition, 2009.

[9] A. Farzaneh, H. Kheiri, and M. A. Shahmersi, "Fundamental study perfect hashing," *Theoretical Computer Science*, vol. 182, no. 1, pp. 1–143, 1997.

[10] A. Farzaneh, H. Kheiri, and M. A. Shahmersi, "An efficient storage format for large sparse matrices," *Commun.Fac.Sci.Univ.Ank.Series A1*, vol. 58, no. 2, pp. 1–10, 2009.

[11] E. A. Fox, L. S. Heath, Q. F. Chen, and A. M. Daoud, "Practical minimal perfect hash functions for large databases," *Communications of the ACM*, vol. 35, no. 1, pp. 105–121, 1992.

[12] M. G. Gouda, A. X. Liu, and M. Jafry, "Verification of distributed firewalls," in *IEEE Global Communications Conference (GLOBECOM'08)*, pp. 1–5, New Orleans, LO, Nov-Dec 2008.

[13] H. Hamed, A. El-Atawy, and E. Al-Shaer, "On dynamic optimization of packet matching in high-speed firewalls," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 10, pp. 1817–1830, 2006.

[14] X. He, T. Chomsiri, P. Nanda, and Z. Tan, "Improving cloud network security using the tree-rule firewall," *Future Generation Computer Systems*, vol. 30, pp. 116–126, 2013.

[15] T. Heinz, "Hipac high performance packet classification for netfilter," 2004. (`http://www.net.t-labs.tu-berlin.de/papers/H-HiPAC-04.pdf`)

[16] E. Horowitz and S. Sahni, *Fundamentals of Data Structures.* Potomac, Maryland: Computer Science Press, 1976.

[17] J. H. Jiang, C. C. Chang, and T. S. Chen, "A compact sparse matrix representation using random hash functions," *Data and Knowledge Engineering*, vol. 32, no. 1, pp. 29–49, 2000.

[18] S. Khummanee, A. Khumseela, and S. Puangpronpitag, "Towards a new design of firewall: Anomaly elimination and fast verifying of firewall rules," in *Proceedings of The Computer Science and Software Engineering (JCSSE'13)*, pp. 93–98, Maha Sarakham, May 2013.

[19] S. Khummanee and K. Tientanopajai, "The policy mapping algorithm for high-speed firewall policy ver-

ifying," *International Journal of Network Security*, vol. 18, no. 3, pp. 433–444, 2016.

[20] A. X. Liu, "Formal verification of firewall policies," in *IEEE International Conference on Communications*, pp. 1494–1498, Beijing, May 2008.

[21] A. X. Liu and M. G. Gouda, "Structured firewall design," *Computer Networks Journal*, vol. 51, no. 4, pp. 1106–1120, 2007.

[22] A. X. Liu and M. G. Gouda, "Diverse firewall design," *IEEE Transaction on Parallel and Distributed Systems*, vol. 19, no. 9, pp. 1237–1251, 2008.

[23] A. X. Liu and M. G. Gouda, "Firewall policy queries," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 6, pp. 766–777, 2009.

[24] Netfilter, "IPSET," 2015. (`http://ipset.netfilter.org/index.html`)

[25] Landon Curt Noll, "Fnv hash," 2015. (`http://isthe.com/chongo/tech/comp/fnv/`) 2015.

[26] S. Pissanetsky, *Sparse Matrix Technology*. San Diego, CA: Academic Press Inc, 1984.

[27] M. Roberts, "Sparse matrix compression formats," 2015. (`http://www.cs.colostate.edu/mcrob/toolbox/cpp/sparseMatrix/sparse-matrix-compression.html`)

[28] D. Rovniagin and A. Wool, "The geometric efficient matching algorithm for firewalls," *IEEE Transactions on Dependable And Secure Computing*, vol. 8, no. 1, pp. 147–159, 2011.

[29] R. Russell, "IPTables," 2015. (`http://ipset.netfilter.org/iptables.man.html`) 2015.

[30] C. A. Shaffer, *A Practical Introduction to Data Structures and Algorithm Analysis Third Edition*. Blacksburg, USA: Virginia Tech, 2010.

[31] J. M. Stewart, *Network Security, Firewalls, And Vpns*. Burlington: Jones and Bartlett Learning, 2014.

[32] J. van Leeuwen, *Handbook of Theoretical Computer Science*. Cambridge MA: The MIT Press, 1990.

[33] Wikipedia, "Sparse matrix," 2015. (`https://en.wikipedia.org/wiki/Sparsematrix`)

Suchart Khummanee is a Ph.D student at Khon Kaen University, Khon Kaen, Thailand. His research is in the field of Computer Networks and Security.

Kitt Tientanopajai is a full lecturer of computer engineering with Khon Kaen University, Khon Kaen, Thailand. His research interests focus on Free/Open Source Software, Information Security, Quality of Service Routing, Computer Networks and Educational Technology.

A Novel Micropayment Scheme with Variable Denomination

Quan-Yu Zhao[1], Yi-Ning Liu[2], Gao Liu[1] and Chin-Chen Chang[3]

(Co-corresponding authors: Yi-Ning Liu, Chin-Chen Chang)

School of Mathematics and Computational Science, Guilin University of Electronic Technology, Guilin 541004, China[1]

Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China[2]

Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan[3]

(Email: ynliu2011@gmail.com, alan3c@gmail.com)

(Received Oct. 19, 2015; revised and accepted Jan. 23 & Mar. 6, 2016)

Abstract

Recently, the research of micropayment systems has attracted a lot of attention in the literature. Because of its special feature that large amount of transactions may occur while each transaction only deals with small value, not only security but also efficiency should be carefully considered in these systems, especially when considering that these systems may be implemented in resource constrained mobile devices. In this paper, a novel lightweight micropayment scheme with variable denomination is proposed. Compared with those existing schemes with fixed denomination, our proposed scheme not only guarantees security goals, but also dramatically reduces the computational cost as well as the storage burden.

Keywords: Computation cost, hash chain, micropayment, storage burden, variable denominations

1 Introduction

With the development of the electronic commerce [10], electronic payments (e-payment) are widely used, which usually involves at least three entities: Customer ($\boldsymbol{C}$) who pays for the goods, Vendor ($\boldsymbol{V}$) who provides the goods, and Bank ($\boldsymbol{B}$) who helps the money transfer between $\boldsymbol{C}$ and $\boldsymbol{V}$ [18].

The e-payment scheme can be classified into macro-payment [5] and micropayment [1, 23]. The transaction value in macro-payment is large enough, its security is guaranteed using complicated cryptographic techniques [16, 17]. However, this is unacceptable for micropayment since the transaction value is tiny, maybe it is less than the transaction cost. Therefore, the main goal of micropayment should be lightweight besides the security requirements such as no forgery and privacy as well.

The PayWord was firstly proposed by Rivest and Shamir in [14], which used the hash chain to replace the public key signature to improve the efficiency [8]. Inspired by these works, a lightweight micropayment scheme was proposed to enhance the fairness [9]. In [12], three schemes denoted as SPayword, UPayword and PPayword were presented to improve the robustness. Moreover, the version for multiple vendors was proposed in [3], and the self-renewal micropayment scheme was designed in [11]. In [21], Yang and Teng employed multiple Payword chains to find the minimum hash chain in transaction. Micropayment scheme presented in [4] is an anonymous fair offline scheme for all entities. Moreover, three micropayment schemes [6, 7, 15] were designed for the application over the mobile network.

The PayWord scheme is generalized into two categories: Micropayment Scheme using a Single-Payword Chain (MSSC), and Micropayment Scheme using Multi-Payword Chain (MSMC). Although each value of a hash chain is assigned with the same denomination, the denomination of each value in one hash chain is different from that of the others. Therefore, $\boldsymbol{C}$ can use the value of a hash chain to purchase in the MSSC scheme, and two hash chains are employed for the transaction in the MSMC scheme. Obviously, it is more efficient to utilize the multi-Payword hash chains [13, 19, 20] to reduce the computation cost.

However, the hash chain with small denomination might be used faster than the one with large denomination in the MSMC scheme, which might result in the storage waste due to the redundancy of the hash value. To make the best use of the redundancy multi-Payword hash chain, MSRC was presented in [22]. In order to reduce the computation and storage burden, a novel micropayment scheme using a hash chain is proposed. Each value in the hash chain is taken weight by adopting a pre-defined function. Moreover, the weight is used as the denomination of this hash value. Therefore, the denominations of the hash chain are variable. As a result, the proposed scheme with variable denomination is more efficient than the schemes with fixed denomination.

The rest of this paper is organized as follows. In

Section 2, we introduce and analyze the MSRC scheme. Moreover, a variable denomination micropayment is proposed in Section 3. In Section 4, we give the analysis of presented scheme. The comparison of the micropayment schemes is given in Section 5. Finally, Section 6 concludes the paper.

2 Yang-Wu's MSRC

2.1 Micropayment Scheme to Return Changes (MSRC)

The MSRC scheme involves three participants, i.e., $\boldsymbol{C}$, $\boldsymbol{B}$ and $\boldsymbol{V}$, their corresponding public key and private key are (PK_C, SK_C), (PK_V, SK_V) and (PK_B, SK_B). Furthermore, $\{\cdot\}_{PK}$ and $\{\cdot\}_{SK}$ denote the encryption and signature using the public key and private key. The MSRC scheme consists of four phases: *Registration Phase*, *Transaction Phase*, *Redemption Phase and Remittance Phase.*

1) Registration Phase

Step 1. $\boldsymbol{C}$ sends a payment request $PR = \{ID_C, ID_V, n, r\}$ to $\boldsymbol{B}$, where ID_C and ID_D are the identities of $\boldsymbol{C}$ and $\boldsymbol{V}$, n and r are the length of payment hash chain and returning change hash chain, respectively.

Step 2. $\boldsymbol{B}$ generates three hash chains $(A-chain, A^{'}-chain, B-chain)$ with the denomination d_A, $d_A^{'}$ and d_B, which satisfy $d_A = d_A^{'} < d_B$,

$$\begin{aligned} A &= A_1 \| A_2 \\ &= (a_0, a_1, \cdots, a_n) \| (a_{n+1}, a_{n+2}, \cdots, a_{n+r}) \\ A^{'} &= A_1^{'} \| A_2^{'} \\ &= (a_0^{'}, a_1^{'}, \cdots, a_n^{'}) \| (a_{n+1}^{'}, a_{n+2}^{'}, \cdots, a_{n+r}^{'}) \\ B &= (b_0, b_1, \cdots, b_n) \end{aligned}$$

where $a_i = h(a_{i+1})$, $a_i^{'} = h(a_{i+1}^{'})$, for $i = n + r - 1, n + r - 2, \cdots, 0$, and $b_j = h(b_{j+1})$, for $i = n - 1, n - 2, \cdots, 0$, $h(\cdot)$ is a secure hash function such as SHA-256, and $\|$ is the concatenation operation.

Step 3. $\boldsymbol{B}$ sends $\{A, A_1^{'}, B\}_{PK_C}$ to $\boldsymbol{C}$, and sends $\{a_0, a_0^{'}, b_0, A_2^{'}\}_{PK_V}$ to $\boldsymbol{V}$.

Step 4. $\boldsymbol{C}$ derives the payment chain $\{A, A_1^{'}, B\}$ by decrypting $\{A, A^{'}, B\}_{PK_C}$ with SK_C.

Step 5. $\boldsymbol{V}$ obtains the root $\{a_0, a_0^{'}, b_0\}$ and the returning chain $A_2^{'}$ by decrypting $\{a_0, a_0^{'}, b_0, A_2^{'}\}_{PK_V}$ with SK_V.

2) Transaction Phase
$\boldsymbol{C}$ uses the e-cash $A - chain$ and $B - chain$ to pay for the goods. For simplicity, an example which the price of the goods is 27 dollars is used to illustrate this phase.

In the MSRC scheme, $d_A = \$1$ and $d_B = \$10$ were assumed. Then $\boldsymbol{C}$ should spend 2 hash values in $B - chain$ and 7 hash values in chain A_1 and $A_1^{'}$. That's to say the hash chain A_1 and $A_1^{'}$ with small denomination will be fetched away soon. When the hash chains A_1 and $A_1^{'}$ are exhausted, $\boldsymbol{C}$ must spend 3 hash values in $B-chain$, and $\boldsymbol{V}$ needs to spend 3 hash values in $A_2^{'}$, which saves the hash values with small denomination. The processes of $\boldsymbol{C}$ pays $M \times d_B$ money and $\boldsymbol{V}$ returns $m \times d_A$ money are shown in the following listing.

Step 1. $\boldsymbol{C}$ sends (b_M, M) to $\boldsymbol{V}$.

Step 2. $\boldsymbol{V}$ checks the equation $b_0 = h^M(b_M)$. If the equation holds, $\boldsymbol{V}$ transmits $(a_{n+m}^{'}, m)$ to $\boldsymbol{C}$ for returning $m \times d_A$ money, provides goods to $\boldsymbol{C}$, and stores b_M instead of b_0.

Step 3. $\boldsymbol{C}$ checks the equation $a_n^{'} = h^m(a_{n+m}^{'})$. If the equation holds, $\boldsymbol{C}$ ensures that $\{a_i, a_i^{'}\}$ $(i = n + 1, n + 2, \cdots, n + m)$ can be used for the payment.

3) Redemption Phase and Remittance Phase
If $\boldsymbol{V}$ wants to redeem the money from $\boldsymbol{B}$, he executes the following steps.

Step 1. $\boldsymbol{V}$ sends $\{ID_C, ID_V, a_{i_1}, a_{i_1}^{'}, b_{j_1}\}_{SK_V}$ to $\boldsymbol{B}$.

Step 2. $\boldsymbol{B}$ verifies $i_1 \le n+r$ and $j_1 \le n$. If i_1 and j_1 are valid, $\boldsymbol{B}$ remits the money $i_1 \times d_A + j_1 \times d_B$ to $\boldsymbol{V}$ and stores $(a_{i_1}, a_{i_1}^{'}, b_{j_1})$ simultaneously.

Step 3. In the next time, $\boldsymbol{B}$ receives the redemption request $\{ID_C, ID_V, a_{i_2}, a_{i_2}^{'}, b_{j_2}\}_{SK_V}$ from $\boldsymbol{V}$. Therefore, $\boldsymbol{B}$ remits the money $(i_2 - i_1) \times d_A + (j_2 - j_1) \times d_B$ to $\boldsymbol{V}$ and stores $(a_{i_2}, a_{i_2}^{'}, b_{j_2})$ when $i_2 \le n + r$ and $j_2 \le n$ hold.

2.2 Analysis of Yang-Wu's MSRC

With the ability of returning change, MSRC scheme is in favor of fabricating the hash chains with the length of the appropriate ratio, then the hash chain with the shortest length is often found to be used for purchasing the same goods. As the PayWord with the small denomination, $(a_i, a_i^{'})$ is utilized for the payment. Moreover, the chain with returning change cannot be used unless the hash chains with the small denomination are exhausted. Before the chain with returning change is used, the $\boldsymbol{V}$'s verification costs twice hash computations, and $\boldsymbol{C}$ needs to store twice hash chains A and $A^{'}$ with the same length. Therefore, the computation and storage burden should be paid attention to.

In order to reduce the computation and storage burden, a variable denomination micropayment scheme is presented using Hamming weight [2], in which only one hash chain is necessary and the denomination of values in it might not be the same.

3 Variable Denomination Micropayment (VDM) Scheme

In VDM scheme, the range of the transaction values is assumed in the set $D = \{1, 2, \cdots, s\}$, where s is the max transaction value. Moreover, the denomination $d_i = WF(a_i) mod P + 1$ is assigned to a_i, where $WF(x)$ is denoted as the weight function with the value of x such as Hamming weight function, or other method that can be defined as the value of some bits, P is a prime number. Then d_i is an positive integer in the range $[1, P]$. In this paper, Hamming weight is assigned to $WF(x)$, the binary representation of 9 is 1001, so $WF(9) = 2$.

The VDM scheme also consists of four phases: *Registration Phase*, *Transaction Phase*, *Redemption Phase and Remittance Phase*. Moreover, the details of each phase are described as follows.

1) Registration Phase
$\boldsymbol{C}$, $\boldsymbol{V}$ select random pseudonym identities ID_C and ID_V which must be different from that of others. The identities of them should be verified by $\boldsymbol{B}$.

Step 1. $\boldsymbol{C}$ sends a payment request $PR = \{ID_C, ID_V, n\}$ to $\boldsymbol{B}$ for getting a payment hash chain.

Step 2. $\boldsymbol{B}$ generates an $n - length$ hash chains,

$$A = (a_0, a_1, \cdots, a_n)$$

where $a_i = h(a_{i+1})$ for $i = n-1, n-2, \cdots, 0$.

Step 3. $\boldsymbol{B}$ sends $\{A\}_{PK_C}$ to $\boldsymbol{C}$, and sends $\{a_0\}_{PK_V}$ to $\boldsymbol{V}$.

Step 4. $\boldsymbol{C}$ derives the payment hash chain A by using the private key SK_C to decrypt $\{A\}_{PK_C}$.

Step 5. $\boldsymbol{V}$ obtains the root a_0 by decrypting $\{a_0\}_{PK_V}$ with SK_V.

2) Transaction Phase

Step 1. Before $\boldsymbol{C}$ sends the first purchase request, he chooses a_{m_1} which the total denominations satisfy $t_1 = \sum_{j=1}^{m_1} d_j - W_1$ and $|t_1| < P$, where W_1 is the true price of the first purchased goods. Moreover, $\boldsymbol{C}$ sends (a_{m_1}, m_1, t_1) to $\boldsymbol{V}$.

Step 2. $\boldsymbol{V}$ computes $t_1^{'} = \sum_{j=1}^{m_1} d_j - W_1$ and checks $t_1^{'} = t_1$; then checks $a_0 = h^{m_1}(a_{m_1})$ and $|t_1^{'}| < P$, If all checks hold, $\boldsymbol{V}$ provides $\boldsymbol{C}$ with the goods and stores (a_{m_1}, t_1).

Step 3. In the $i - th$ transaction, $\boldsymbol{C}$ selects $a_{m_1^{'}}$ which $t_{i-1}t_i < 0$ (where $t_i = t_{i-1} + \sum_{j=m_{i-1}^{'}+1}^{m_i^{'}} d_j - W_i$ $(i \geq 2)$) and $|t_i| < P$. Then $\boldsymbol{C}$ sends $(a_{m_i^{'}}, m_i^{'} - m_{i-1}^{'}, t_i)$ to $\boldsymbol{V}$, where t_i is the error between the paid money and the true price $\sum_{j=1}^{i} W_j$ of the purchased goods for the previous i times transactions, and $m_i^{'} - m_{i-1}^{'}$ is the number of hash operations in the $i - th$ transaction.

Step 4. $\boldsymbol{V}$ computes $t_i^{'} = t_{i-1}^{'} + \sum_{j=m_{i-1}^{'}+1}^{m_i^{'}} d_j - W_i$, provides $\boldsymbol{C}$ with the goods and stores $(a_{m_i^{'}}, t_i)$ if all of $t_i^{'} = t_i$, $a_{m_{i-1}^{'}} = h^{m_i^{'} - m_{i-1}^{'}}(a_{m_i^{'}})$ and $|t_i^{'}| < P$ are hold.

3) Redemption Phase and Remittance Phase

Step 1. $\boldsymbol{V}$ sends $\{ID_C, ID_V, i_1, a_{i_1}\}_{PK_B}$ to $\boldsymbol{B}$.

Step 2. $\boldsymbol{B}$ derives $\{ID_C, ID_V, i_1, a_{i_1}\}$ by using the private key SK_B. Furthermore, if $i_1 \leq n$ is valid, $\boldsymbol{B}$ recharges the money $Q_{i_1} = \sum_{j=1}^{i_1} d_j$ to $\boldsymbol{V}$ and stores a_{i_1}, where Q_i is the money $\boldsymbol{B}$ remits to $\boldsymbol{V}$ in the $i - th$ transaction.

Step 3. When $\boldsymbol{B}$ receives $\{ID_C, ID_V, i_2, a_{i_2}\}_{PK_B}$ in the next remittance period, where $i_2 \leq n$, then $\boldsymbol{B}$ calculates $Q_{i_2} = \sum_{j=i_1+1}^{i_2} d_j$, deposits the money Q_{i_2} to $\boldsymbol{V}$ and replaces a_{i_1} with a_{i_2}.

4 System Analysis

In this section, we analyze the security and efficiency of the proposed scheme and compare it with other schemes.

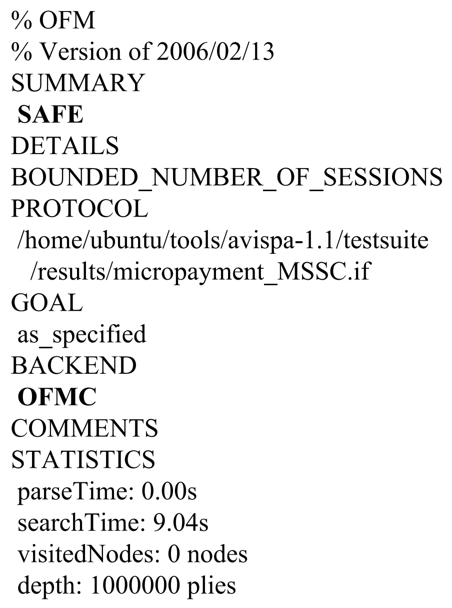
4.1 Security Analysis

The proposed micropayment scheme achieves the correctness, anonymity, un-traceability, fairness, and unforgeability.

Correctness.
In the VDM scheme, $|t_i| < P$ is checked by $\boldsymbol{V}$ in each transaction. Furthermore, the error rate between the error value t_i and the total of true price in the previous i times transactions is defined as $E_i = \frac{|t_i|}{\sum_{j=1}^{i} W_j} \times 100\%$. Obviously, E_i will reduce when the transaction amount increases. As a result, t_i is acceptable for $\boldsymbol{V}$ and $\boldsymbol{C}$ since the transaction amount is huge.

The AVISPA tool can analyze the security of the protocols, and check whether the scheme is safe or unsafe against the passive and active adversaries. Moreover, the AVISPA tool is used for the automated validation of security-sensitive protocols and applications. The simulation results are showed in Figure 1. From the results, it is evident that our scheme is secure against attack, and satisfies the confidentiality.

Anonymity and Un-traceability.
In the proposed scheme, $\boldsymbol{C}$ and $\boldsymbol{V}$ randomly select the pseudonym identities ID_C and ID_V , which need not have any relationship with the real identities of them, and must be different from that of any others. Therefore, nobody including $\boldsymbol{C}$, $\boldsymbol{V}$ and $\boldsymbol{B}$ can trace the real identity of the entities via the data exchange. Then the VDM scheme satisfies the anonymous and the un-traceability.



```
% OFM
% Version of 2006/02/13
SUMMARY
 SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
 /home/ubuntu/tools/avispa-1.1/testsuite
   /results/micropayment_MSSC.if
GOAL
 as_specified
BACKEND
 OFMC
COMMENTS
STATISTICS
 parseTime: 0.00s
 searchTime: 9.04s
 visitedNodes: 0 nodes
 depth: 1000000 plies
```

Figure 1: The simulation results of analysis using AVISPA

Fairness.

If the scheme is executed functionally, it is fair for all entities. The scheme adopts the prepaid method in the transaction, $\boldsymbol{C}$ must pay the money to $\boldsymbol{B}$ before sending the payment request. Therefore, the profit of $\boldsymbol{B}$ and $\boldsymbol{V}$ are protected. Moreover, $\boldsymbol{B}$ is responsible for creating the hash chain, so the risk of the $\boldsymbol{C}$'s overspending is eliminated. Furthermore, $\boldsymbol{C}$ pays the money to $\boldsymbol{V}$ after receiving the goods, so he takes no risk of unfairness.

Unforgeability.

Security plays a vital and central role in the e-commerce payment scheme. In this subsection, we analyze that the VDM scheme resists against the outsider and insider forgery attacks, it means that no forgery attacks exist in the proposed scheme.

No forgery attacks mean an attacker cannot fake the value of hash chain for payment by eavesdropping the information from the public channel.

Scenario 1. *It is unfeasible for an attacker to forge the hash chain to exchange money if the protocol is processed functionally.*

Proof. Primarily, the external malicious attacker cannot forge and spend the hash chain since hash chain is a strong cryptography one-way function and the information transferred among $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{V}$ is always encrypted by a public key. Meanwhile, the external malicious attacker obtains the information $(a_{m_i'}, m_i' - m_{i-1}', t_i)$ from the public channel, he cannot derive $(a_{m_i'+1}, a_{m_i'+2}, \cdots, a_n)$ to redeem money from $\boldsymbol{B}$ due to the feature of secure hash function. Moreover, the external attacker cannot obtain a_{i_1} from the eavesdropped $\{ID_C, ID_V, i_1, a_{i_1}\}_{PK_B}$. The external malicious attacker cannot use the identity of $\boldsymbol{V}$ to exchange money for himself, and the money can be remitted if and only if the identity of $\boldsymbol{V}$ can be verified. Therefore, the attacker cannot forge the hash chain for payment or obtain the illegal benefit from $\boldsymbol{B}$.

Secondly, the forgery attack from the internal $\boldsymbol{C}$ means that $\boldsymbol{C}$ attempts to use the hash chains to double spend or to overspend hash value. In the proposed scheme, $\boldsymbol{C}$ cannot spend more than the total money of the hash chain since he should pay the money to $\boldsymbol{B}$ in advance. Simultaneously, $\boldsymbol{B}$ will track the balance of the total money of the hash chain and avoid the overspending hash chain values. Moreover, the used hash chain will be destroyed by $\boldsymbol{B}$ and will not be allowed to double spend. Obviously, $\boldsymbol{C}$ cannot launch this attack.

Finally, the forgery attack from the internal $\boldsymbol{V}$ implies that $\boldsymbol{V}$ sends $\{ID_C, ID_V, i_1, a_{i_1}\}_{PK_B}$ to $\boldsymbol{B}$ for exchanging more money. If $i_1 \leq n$, $\boldsymbol{V}$ can redeem money Q_{i_1} from $\boldsymbol{B}$, but he cannot get a_i $(i > i_1)$. Therefore, it is impossible for the internal $\boldsymbol{V}$ to redeem more money from $\boldsymbol{B}$. If $\boldsymbol{V}$ receives a_i $(i \leq n)$, he cannot obtain a_j $(j > n)$ to redeem money. Hence, he cannot use the received a_i to redeem the illegal money from $\boldsymbol{B}$. □

4.2 Efficiency Analysis

The $\boldsymbol{C}$, $\boldsymbol{V}$ and $\boldsymbol{B}$ involved in the proposed VDM only needed their (PK_C, SK_C), (PK_V, SK_V) and (PK_B, SK_B). In the registration phase, the private key computation SK_C is executed once by $\boldsymbol{C}$ and the private key computation SK_V is carried out once by $\boldsymbol{V}$. Moreover, $\boldsymbol{B}$ executes the corresponding public key computation PK_C and PK_V once. However, the private key computation SK_B is executed once by $\boldsymbol{B}$ and the corresponding public key computation PK_B is used once by $\boldsymbol{V}$ in each redemption phase and remittance phase. No certificates are needed in the transaction. Moreover, the information stored by the entities is much more simple and entities can be implemented the protocol as soon as possible. Compared with the other schemes which used the complicated algorithm, the proposed scheme is highly efficient.

5 Comparison

In this subsection, some important micropayment schemes (MSSC, MSMC, MSRC and VDM) are further discussed: (1) the number of hash and encryption operation, (2) the storage burden. The comparison of the computation and storage burden among the MSSC, the MSMC, the MSRC and the proposed schemes will be summarized. Furthermore, only the hash function computation is considered since the complicated computations such as the signature and encryption operations are the same in these schemes. In addition, $d = \frac{d_B}{d_A}$ is defined due to the two chains with different denomination d_A and d_B, where $d_A < d_B$ in all micropayment schemes.

The computation cost in MSRC scheme is approximate with that of MSMC scheme. Afterwards, the computation cost in VDM scheme is much smaller than that of the above schemes. Therefore, the computation costs of MSSC, MSRC schemes are compared with that of our VDM scheme.

The error rate E_i is controlled within 5%, and the hash operation amount of previous s transactions in MSSC, MSRC schemes are compared with that of the VDM scheme, which is shown in Figure 2.

As depicted in Figure 2, the computation cost in VDM scheme is significant smaller than that of others. We compare the storage burden requirement of other schemes with VDM scheme in Table 1. Obviously, our storage burden is also lighter than that of the other schemes.

6 Conclusions

By using the Hamming weight function in micropayment scheme, the VDM scheme is presented to reduce the hash operations in the transaction phases. Each hash value in VDM scheme is taken weight by adopting the Hamming weight function. Moreover, the weight is used as the denomination of this hash value. Therefore, the denominations of the hash chain are variable. Compared with the fixed denomination schemes, the proposed scheme not only ensures the privacy, anonymity, un-traceability, fairness, unforgeability and authentication, but also is more efficient with respect to the computation and storage burden. Therefore, the presented scheme is suitable for the secure mobile devices communication.

Acknowledgments

Project supported by the Natural Science Foundation of China (No. 61363069, 61301166), Guangxi Natural Science Foundation (No. 2014GXNSFAA118364), Innovation Project of Guangxi Graduate Education (No. YCSZ2013069), the High Level Innovation Team of Guangxi Colleges and Universities, and the Program for Innovative Research Team of Guilin University of Electronic Technology.

References

[1] V. Daza and F. Lombardi, "Frodo: Fraud resilient device for off-line micropayment," *IEEE Transactions on Dependable and Secure Computing*, DOI: 10.1109/TDSC.2015.2432813.

[2] C. Ding and J. Yang, "Hamming weights in irreducible cyclic codes," *Discrete Mathematics*, vol. 313, no. 4, pp. 434–446, 2013.

[3] A. Esmaeeli and M. Shajari, "Mvpayword: Secure and efficient payword-based micropayment scheme," in *Applications of Digital Information and Web Technologies*, pp. 609–614, 2009.

[4] C. Fan, Y. Liang, and C. Wu, "An anonymous fair offline micropayment scheme," in *International Conference on Information Society*, pp. 377–381, 2011.

[5] T. Fun, L. Beng, and M. Razali, "Review of mobile macro-payment schemes," *Journal of Advances in Computer Networks*, vol. 1, pp. 323–328, 2013.

[6] A. Isern-Dey, M. Payeras-Capell, and M. Mut-Puigserver, "Micropayment proposal with formal verification using coloured petri nets and performance analysis on the android platform," *International Journal of Business Intelligence and Data Mining*, vol. 8, no. 1, pp. 74–104, 2013.

[7] N. Kiran and G. Kumar, "Implication of secure micropayment system using process oriented structural design by hash chaining in mobile network," *International Journal of Computer Science Issues*, vol. 9, pp. 329–339, 2012.

[8] Y. Liu, L. Hu, and H. Liu, "A micropayment scheme based on weighted multi-dimensional hash chain," *Journal of Electronics (China)*, vol. 23, pp. 791–794, 2006.

[9] Y. Liu and J. Yan, "A lightweight micropayment scheme based on lagrange interpolation formula," *Security and Communication Networks*, vol. 6, pp. 995–960, 2013.

[10] J. Lo, H. Lu, T. Sun, and M. Hwang, "Improved on date attachable electronic cash," *Applied Mechanics and Materials*, vol. 284-287, pp. 3444–3448, 2013.

[11] J. Meng and Y. Yang, "Self-renewal hash chains micropayment protocol based on payword," *Computer Engineering*, vol. 35, pp. 63–65, 2009.

[12] Y. Mu, V. Varadharajan, and Y. Lin, "New micropayment schemes based on paywords," *Lecture Notes in Computer Science*, vol. 1270, pp. 283–293, Springer, 1997.

[13] S. Quan, "Multi-dimensional hash chains and application to micropayment schemes," *Lecture Notes in Computer Science*, vol. 3969, pp. 218–228, Springer, 2006.

[14] R. Rivest and A. Shamir, "Payword and micromint: Two simple micropayment schemes," *Lecture Notes in Computer Science*, vol. 1189, pp. 69–87, Springer, 1997.

[15] L. Rotger, M. Capell, and M. Puigserver, "Anonymous, fair and untraceable micropayment scheme: Application to LBS," *IEEE Latin America Transactions*, vol. 10, pp. 1774–1784, 2012.

[16] K. R. Santosh, C Narasimham, and P Shetty, "Cryptanalysis of multi-prime rsa with two decryption exponents," *International Journal of Electronics and Information Engineering*, vol. 4, no. 1, pp. 40–44, 2016.

[17] G Sharma, S Bala, and A. Verma, "An improved rsa-based certificateless signature scheme for wireless sensor networks," *International Journal of Network Security*, vol. 18, no. 1, pp. 82–89, 2016.

[18] M. Silvio and L. Ronald, "Micropayment revisited," *Lecture Notes in Computer Science*, vol. 2271, pp. 149–163, Springer, 2002.

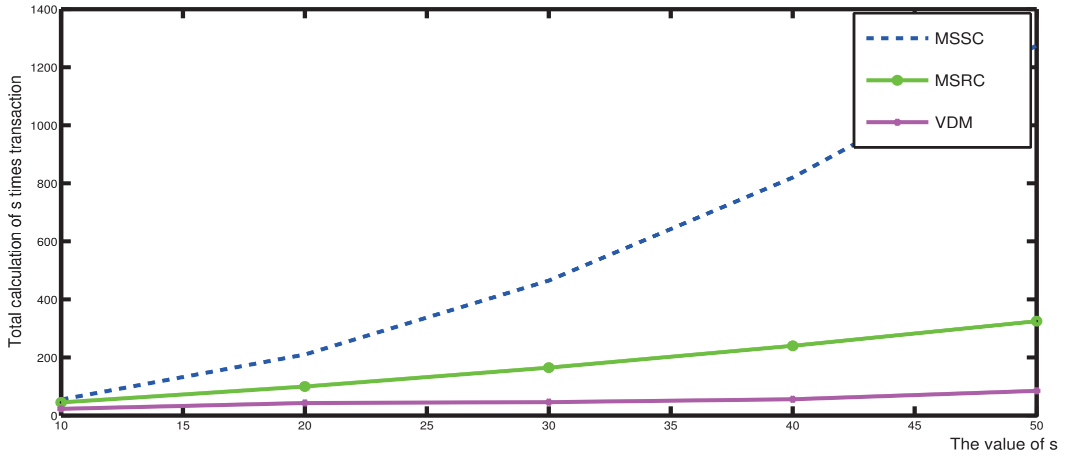


Figure 2: Total hash operation comparison of s times transaction

Table 1: Storage burden comparison of related works with ours

	MSSC	**MSMC**	**MSRC-I**	**MSRC-II**	**MSRC-III**	**VDM**
Encryption key	No	No	$\boldsymbol{C}$ need	No	$\boldsymbol{C}$ and $\boldsymbol{V}$ need	No
$\boldsymbol{C}$*'s storage burden*	$(d+1)n$	$2n$	$2n$	$3n+r$	$2n+1$	$\frac{22n}{1+P}$
$\boldsymbol{V}$*'s storage burden*	1	2	$r+2$	$r+3$	$\geq r+2$	2
Total storage burden	$(d+1)n+1$	$2n+2$	$2n+r+2$	$3n+2r+3$	$\geq 2n+r+3$	$\frac{22n}{1+P}+2$

[19] H. Wang, J. Ma, and J. Sun, "Micro-payment protocol based on multiple hash chains," in *Second International Symposium on Electronic Commerce and Security*, pp. 71–74, 2009.

[20] H. Wang, Z. Wang, and Y. Bo, "Micropayment system based on self-updatable two-dimensional hash chain," *Computer Engineering*, vol. 37, no. 18, pp. 272–274, 2011.

[21] C. Yang and H. Teng, "An efficient method for finding minimum hash chain of multi-payword chains in micropayment," in *IEEE International Conference on E-Commerce*, pp. 45–48, 2003.

[22] C. Yang and C. Wu, "Msrc: Micropayment scheme with ability to return changes," *Mathematical and Computer Modelling*, vol. 58, no. 1-2, pp. 96–107, 2013.

[23] S. Yen, H. Lin, Y. Chen, J. Hung, and J. Wu, "Paystar: A denomination flexible micropayment scheme," *Information Sciences*, vol. 259, pp. 160–169, 2014.

Quan-yu Zhao is currently pursuing his M.S. degree in Guilin University of Electronic Technology, Guilin, China. He received the B.S. degree in Mathematics and Applied Mathematics from Huainan Normal University, Anhui, China, in 2014. His research interests focus on e-voting, micropayment, e-lottery, group key transfer, and oblivious transfer.

Yi-Ning Liu is currently a professor in Guilin University of Electronic Technology, Guilin, China. He received the B.S. degree in Applied Mathematics from Information Engineering University, Zhengzhou, China, in 1995, the M.S. in Computer Software and Theory from Huazhong University of Science and Technology, Wuhan, China, in 2003, and the Ph.D. degree in Mathematics from Hubei University, Wuhan, China, in 2007. His research interests include the analysis of information security protocol, the smart grid, and e-voting.

Gao Liu is now pursuing his M.S. degree in Guilin University of Electronic Technology, Guilin, China. He received the B.S. degree in Applied Mathematics from Yibin University, Sichuan, China, in 2013. His research interests focus on e-voting, micropayment, e-lottery, and smart grid.

Chin-Chen Chang received his Ph.D. degree in computer engineering from National Chiao Tung University. His first degree is Bachelor of Science in Applied Mathematics and master degree is Master of Science in computer and decision sciences. Both were awarded in National Tsing Hua University. Dr. Chang served in National Chung Cheng University from 1989 to 2005. His title is Chair Professor in Department of Information Engineering and Computer Science, Feng Chia University, from Feb. 2005. He is a Fellow of IEEE and a Fellow of IEE, UK. His research interests include database design, computer cryptography, image compression and data structures.

The Paillier's Cryptosystem and Some Variants Revisited

Zhengjun Cao[1], Lihua Liu[2]
(Corresponding author: Lihua Liu)

Department of Mathematics, Shanghai University[1]
No. 99, Shangda Road, Shanghai, China
Department of Mathematics, Shanghai Maritime University[2]
No. 1550, Haigang Ave, Pudong New District, Shanghai, China
(Email: caozhj@shu.edu.cn; liulh@shmtu.edu.cn)

(Received Dec. 17, 2015; revised and accepted Mar. 6, 2016)

Abstract

At Eurocrypt'99, Paillier presented a public-key cryptosystem based on a novel computational problem. It has interested many researchers because of its additively homomorphic property. In this paper, we show that there is a big difference between the original Paillier's encryption and some variants. The Paillier's encryption can be naturally converted into a signature scheme but these variants miss the feature. In particular, we simplify the alternative decryption procedure of Bresson-Catalano-Pointcheval encryption scheme proposed at Asiacrypt'03. The new version is more applicable to cloud computing because of its double trapdoor decryption mechanism and its flexibility to be integrated into other cryptographic schemes. It captures a new feature that its two groups of secret keys can be distributed to different users so as to enhance the robustness of key management.

Keywords: Additively homomorphic encryption, double trapdoor decryption, Paillier's cryptosystem, robustness of key management

1 Introduction

Homomorphic encryption is a useful cryptographic primitive because it can translate an operation on ciphertexts into an operation on underlying plaintexts.

The property is very important for many applications, such as e-voting, threshold cryptosystems, watermarking and secret sharing schemes. For example, if an additively homomorphic encryption is used in an e-voting scheme, one can obtain an encryption of the sum of all ballots from their encryption. Consequently, it becomes possible that a single decryption will reveal the result of the election. That is, it is unnecessary to decrypt all ciphertexts one by one.

At Eurocrypt'99, Paillier [21] proposed a public-key cryptosystem based on a novel computational problem. It encrypts a message m by

$$E(m, r) = g^m r^n \mod n^2,$$

where $n = pq$ is an RSA modulus, g is a public parameter such that $n \,|\, \mathrm{ord}_{n^2}(g)$, and r is a random pad. The encryption function $E(m, r)$ has the additively homomorphic property, i.e.,

$$E(m_1, r_1)E(m_2, r_2) = E(m_1 + m_2, r_1 r_2).$$

More powerful, who knows the trapdoor of the encryption function can recover not only the message m but also the random pad r. This is another appreciated property for many applications. Due to this property, the Paillier's encryption scheme can be naturally transformed into a one-way trapdoor permutation and a digital signature scheme.

In 1984, Goldwasser and Micali [13] proposed the first probabilistic encryption scheme which was also homomorphic. It has been improved [19, 20]. In 1999, Paillier [21] presented a novel additively homomorphic encryption which was more powerful because it can recover the random pad r as well as the message m. At PKC'01, Damgård and Jurik [9] put forth a generalization of Paillier's encryption using computations modulo $n^i (i \geq 2)$ and taking a special base $g = n + 1$.

They [10] also investigated the applications of the generalization. The elliptic curve variant of Paillier's cryptosystem is due to Galbraith [11].

In 2001, Choi et al. [7] revisited the Paillier's encryption by taking a special base g such that $g^\lambda = 1 + n \mod n^2$, where $\lambda = \mathrm{lcm}(p-1, q-1)$. Shortly after that, Sakurai and Takagi [22] pointed out that the variant cannot resist a chosen ciphertext attack which can factor the modulus n by only one query to the decryption oracle.

At Eurocrypt'06, Schoenmakers and Tuyls [23] have considered the problem of converting a given Paillier's encryption of a value $x \in \mathbb{Z}_n$ into Paillier's encryption of the bits of x. At Eurocrypt'13, Joye and Libert [15] obtained another generalization based on 2^k-th power residue problem. In 2013, Boneh et al. [1] considered the problem of private database queries using Paillier's homomorphic encryption. At Asiacrypt' 14, Catalano et al. [5] presented an instantiation of publicly verifiable delegation of computation on outsourced ciphertext which supports Paillier's encryption. In 2015, Castagnos and Laguillaumie [4] designed a linearly homomorphic encryption scheme whose security relies on the hardness of the decisional Diffie-Hellman problem. Their scheme is somehow similar to the one of [2]. The Gentry's fully homomorphic encryption scheme [12] relies on hard problems related to lattices, which actually allows to evaluate any function on messages given their ciphertexts. But Paillier's cryptosystem based on the problem of factoring RSA integers is still more competitive for applications that need only to add ciphertexts. Recently, Hsien et al. have investigated the possible usage of homomorphic encryption in client-server scenario [6, 14, 16, 17, 18]. Note that a misapplication of a homomorphic encryption for numerical calculations can give rise to errors like the ones in [24] (see [3] for details).

In this paper, we revisit the Paillier's cryptosystem and reaffirm that the Paillier's encryption can be naturally converted into a signature scheme but some variants miss the feature. Our presentation of the cryptosystem and some variants is so plain and heuristic that it becomes possible to investigate the further applications of these schemes in different scenarios. In particular, we simplify the alternative decryption procedure of Bresson-Catalano-Pointcheval encryption scheme. Our new proposal is more applicable to cloud computing because of its double trapdoor decryption mechanism and its flexibility to be integrated into other cryptographic schemes.

It captures a new feature that its two groups of secret parameters can be allocated to different users so as to enhance the robustness of key management.

2 Paillier's Encryption Scheme

Let $n = pq$ be an RSA modulus and $\phi(n)$ be the Euler's totient function. Set $\lambda = \mathrm{lcm}(p-1, q-1)$. Hence, $|\mathbb{Z}_{n^2}^*| = \phi(n^2) = n\phi(n)$ and for any $w \in \mathbb{Z}_{n^2}^*$

$$w^\lambda = 1 \mod n, \quad w^{n\lambda} = 1 \mod n^2$$

which are due to Carmichael's theorem.

Definition 1. *A number z is said to be a n-th residue modulo n^2 if there exists a number $y \in \mathbb{Z}_{n^2}^*$ such that $z = y^n \bmod n^2$.*

The set of n-th residues is a multiplicative subgroup of $\mathbb{Z}_{n^2}^*$ of order $\phi(n)$. Each n-th residue has exactly n roots, among which exactly one is strictly smaller than n.

Let g be some element of $\mathbb{Z}_{n^2}^*$ and define the following integer-valued function

$$\mathcal{E}_g: \quad \mathbb{Z}_n \times \mathbb{Z}_n^* \longmapsto \mathbb{Z}_{n^2}^*$$

$$(x, y) \longmapsto g^x \cdot y^n \mod n^2.$$

Lemma 1. *If $n \mid ord_{n^2}(g)$, then $\mathcal{E}_g$ is bijective.*

Proof. Since the two groups $\mathbb{Z}_n \times \mathbb{Z}_n^*$ and $\mathbb{Z}_{n^2}^*$ have the same number of elements $n\phi(n)$, it suffices to prove that $\mathcal{E}_g$ is injective.

Suppose that $g^{x_1}y_1^n = g^{x_2}y_2^n \bmod n^2$, where $x_1, x_2 \in \mathbb{Z}_n, y_1, y_2 \in \mathbb{Z}_n^*$. It comes $g^{x_2-x_1}(y_2/y_1)^n = 1 \bmod n^2$, which implies

$$\begin{aligned} g^{\lambda(x_2-x_1)}(y_2/y_1)^{\lambda n} &= g^{\lambda(x_2-x_1)} \\ &= 1 \mod n^2. \end{aligned}$$

Thus $\mathrm{ord}_{n^2}(g) \mid \lambda(x_2 - x_1)$.

Since $n \mid \mathrm{ord}_{n^2}(g)$, we have $n \mid \lambda(x_2 - x_1)$. In view of that $(n, \lambda) = 1$, we obtain $x_2 = x_1 \bmod n$. Since $x_1, x_2 \in \mathbb{Z}_n$, it comes $x_1 = x_2$. Thus, $(y_2/y_1)^n = 1 \bmod n^2$, which leads to the unique solution $y_2/y_1 = 1$ over $\mathbb{Z}_n^*$. This means $x_1 = x_2$ and $y_1 = y_2$. Therefore, $\mathcal{E}_g$ is bijective. □

By the above lemma, for a given $w \in \mathbb{Z}_{n^2}^*$, there exists a pair (x, y) such that $w = g^x y^n \bmod n^2$.

Problem 1. *Given an RSA modulus $n = pq$, $c, g \in \mathbb{Z}_{n^2}^*$, compute $x \in \mathbb{Z}_n^*$ such that*

$$g^x y^n = c \mod n^2,$$

where $n \mid ord_{n^2}(g)$ and y is some element of $\mathbb{Z}_{n^2}^*$.

Theorem 1. *If λ is known and $(\frac{g^\lambda - 1 \bmod n^2}{n}, n) = 1$, then one can solve Problem 1 by computing*

$$x = \left(\frac{c^\lambda - 1 \mod n^2}{n}\right)\left(\frac{g^\lambda - 1 \mod n^2}{n}\right)^{-1} \mod n.$$

Proof. By the definition of λ, we have

$$c^\lambda = 1 \mod n, g^\lambda = 1 \mod n.$$

Set

$$c^\lambda = an + 1 \mod n^2, \quad g^\lambda = bn + 1 \mod n^2,$$

i.e.,

$$a = \frac{c^\lambda - 1 \mod n^2}{n}, \quad b = \frac{g^\lambda - 1 \mod n^2}{n}.$$

Since $n \mid \mathrm{ord}_{n^2}(g)$, $\mathcal{E}_g$ is bijective. There exists a pair $(x, y) \in \mathbb{Z}_n \times \mathbb{Z}_n^*$ such that $c = g^x y^n \bmod n^2$. Hence, $c^\lambda = (g^x y^n)^\lambda \bmod n^2$. Since $y^{n\lambda} = 1 \bmod n^2$, it comes $c^\lambda = (g^\lambda)^x \bmod n^2$. Thus,

$$an + 1 = (bn + 1)^x = xbn + 1 \mod n^2$$

this is due to $n^2 \mid \binom{x}{i}(bn)^i, i \geq 2$. Therefore, $an = xbn \bmod n^2$. That means $a = xb \bmod n$. Since $(b, n) = 1$, it gives $x = ab^{-1} \bmod n$. □

Remark 1. *Paillier called the Problem 1 as Composite Residuosity Class Problem (see Definition 8 in [21]). In view of that the trapdoor λ plays a key role in computing the exponent x with respect to the base g, we would like to call the Problem 1 as Trapdoored Partial Discrete Logarithm Problem.*

Conjecture 1. *If the trapdoor λ is unknown, there exists no probabilistic polynomial time algorithm that solves Problem 1.*

Based on the above results, at Eurocrypt'99 Paillier proposed his cryptosystem. The cryptosystem includes a probabilistic encryption scheme, a one-way trapdoor permutation and a digital signature scheme. We now describe the encryption scheme as follows.

Table 1: Paillier's encryption scheme

Setup	Pick an RSA modulus $n = pq$. Set $\lambda = \mathrm{lcm}(p-1, q-1)$. Select $g \in \mathbb{Z}^*_{n^2}$ such that $n \mid \mathrm{ord}_{n^2}(g)$. Publish n, g and keep λ in secret.
Enc.	For $m \in \mathbb{Z}_n$, pick $r \in \mathbb{Z}_n$, compute the ciphertext $c = g^m r^n \bmod n^2$.
Dec.	$m = \left(\frac{c^\lambda - 1 \bmod n^2}{n}\right) / \left(\frac{g^\lambda - 1 \bmod n^2}{n}\right) \bmod n$

3 A Hybrid Computational Problem

We now consider another computational problem which is a hybrid of partial discrete logarithm problem and n-th residuosity problem.

Problem 2. *Given an RSA modulus $n = pq$, $c, g \in \mathbb{Z}^*_{n^2}$, compute $(x, y) \in \mathbb{Z}_n \times \mathbb{Z}^*_n$ such that*

$$g^x y^n = c \bmod n^2$$

where $n \mid ord_{n^2}(g)$.

Notice that the solvability of Problem 2 directly implies that of Problem 1. We shall prove that the inverse holds, too.

If the trapdoor λ is known, Paillier proposed a method to solve the hybrid computational problem. He pointed out that x, y can be computed by

$$\begin{aligned} x &= \left(\frac{c^\lambda - 1 \bmod n^2}{n}\right)\left(\frac{g^\lambda - 1 \bmod n^2}{n}\right)^{-1} \bmod n, \\ y &= (cg^{-x})^{1/n \bmod \lambda} \bmod n. \end{aligned}$$

The idea behind his method can be described as follows. By the existence of (x, y), it is easy to find that

$$\begin{aligned} & g^x y^n = c \bmod n^2 \\ \Longrightarrow\ & g^x y^n = c \bmod n \Longleftrightarrow y^n = cg^{-x} \bmod n \\ \Longleftrightarrow\ & (y^n)^{1/n \bmod \lambda} = (cg^{-x})^{1/n \bmod \lambda} \bmod n \\ \Longleftrightarrow\ & y = (cg^{-x})^{1/n \bmod \lambda} \bmod n \end{aligned}$$

By the uniqueness of $(x, y) \in \mathbb{Z}_n \times \mathbb{Z}^*_n$, we conclude that it is properly computed.

Theorem 2. *If λ is known and $(\frac{g^\lambda - 1 \bmod n^2}{n}, n) = 1$, then one can solve Problem 2 by computing*

$$\begin{aligned} x &= \left(\frac{c^\lambda - 1 \bmod n^2}{n}\right)\left(\frac{g^\lambda - 1 \bmod n^2}{n}\right)^{-1} \bmod n, \\ y &= (cg^{-x})^s \bmod n, \end{aligned}$$

where s is the integer with the least absolute value such that $\lambda \mid ns - 1$.

Proof. Since $(n, \lambda) = 1$, it is easy to compute the integer s with the least absolute value such that $\lambda \mid ns - 1$. By Theorem 1, we conclude that x is properly computed. By the existence of y and $y^n = cg^{-x} \bmod n^2$, we have

$$(cg^{-x})^\lambda = y^{n\lambda} = 1 \bmod n^2$$

Now, suppose that $ns - 1 = \lambda\phi$ and $(cg^{-x})^s = \ell n + y \bmod n^2$ for some integers ϕ, ℓ. Hence, it comes

$$\begin{aligned} g^x \left((cg^{-x})^s - \ell n\right)^n &= g^x (cg^{-x})(cg^{-x})^{ns-1} = c(cg^{-x})^{\lambda\phi} \\ &= c((cg^{-x})^\lambda)^\phi = c \cdot 1^\phi = c \bmod n^2 \end{aligned}$$

This completes the proof. □

Note that the values s and $\left(\frac{g^\lambda - 1 \bmod n^2}{n}\right)^{-1} \bmod n$ have no relation to the ciphertext c. They can be computed and stored previously.

Conjecture 2. *If λ is unknown, there exists no probabilistic polynomial time algorithm that solves Problem 2.*

4 The Paillier's One-way Trapdoor Permutation and The Digital Signature Scheme

In [21], Paillier has put forth a one-way trapdoor permutation and the digital signature scheme based on his computational method. We now relate them as follows.

5 Some Variants of Paillier's Encryption Scheme

5.1 Descriptons of Some Variants

In the same article [21], Paillier has pointed out that there was an efficient variant of his original encryption scheme.

Table 2: Paillier's signature scheme

Setup	See Table 1.
Sign	For a message m, compute $s_1 \leftarrow \rho\left(\frac{H(m)^\lambda - 1 \bmod n^2}{n}\right) \bmod n$. $s_2 \leftarrow ((H(m)g^{-s_1})^s \bmod n$ The signature is $(m; s_1, s_2)$.
Verify	$H(m) \overset{?}{=} g^{s_1} s_2^n \bmod n^2$

Table 3: Paillier's one-way trapdoor permutation

Setup	Set $n = pq$, $\lambda = \mathrm{lcm}(p-1, q-1)$. Select $g \in \mathbb{Z}_{n^2}^*$ such that $n \,\vert\, \mathrm{ord}_{n^2}(g)$. Compute $\rho = \left(\frac{g^\lambda - 1 \bmod n^2}{n}\right)^{-1} \bmod n$, and s which is the integer with the least absolute value such that $\lambda \,\vert\, ns - 1$. Publish n, g and keep λ, ρ, s in secret.
Enc.	Given $m \in \mathbb{Z}_{n^2}$, set $m = m_1 + nm_2$. The ciphertext is $c \leftarrow g^{m_1} m_2^n \bmod n^2$.
Dec.	$m_1 \leftarrow \rho\left(\frac{c^\lambda - 1 \bmod n^2}{n}\right) \bmod n$, $m_2 \leftarrow (cg^{-m_1})^s \bmod n$. $m \leftarrow m_1 + nm_2$.

Shortly afterwards, other variants came out [2, 7, 8, 9]. We list some variants in Table 4.

Correctness of Variant 1. The variant takes $x = m, y = g^r$ in Problem 2. Since each n-th residue has exactly n roots, among which exactly one is strictly smaller than n, and $\mathrm{ord}_{n^2}(g) = \alpha n$, we have $g^\alpha = 1 \bmod n$. Otherwise, suppose $g^\alpha = sn + t \bmod n^2$ for some integers $0 \le s < n$ and $t\,(2 \le t < n)$. It leads to

$$1 = g^{\alpha n} = (sn+t)^n = t^n \mod n^2$$

which means $t = 1$. It is a contradiction. Thus, $g^\alpha = sn + 1 \bmod n^2$. By

$$\begin{aligned} c^\alpha &= (g^m (g^r)^n)^\alpha = (g^\alpha)^m \\ &= (sn+1)^m = smn + 1 \mod n^2 \end{aligned}$$

we have

$$\frac{\frac{c^\alpha - 1 \mod n^2}{n}}{\frac{g^\alpha - 1 \mod n^2}{n}} = \frac{sm}{s} = m \quad \mod n.$$

Correctness of Variant 2. The variant takes $g = 1 + n, x = m, y = r$ in Problem 2. It is easy to find that

$$\begin{aligned} \frac{c^\kappa - 1 \mod n^2}{n} &= \frac{((1+n)^m r^n)^{\tau\lambda} - 1 \mod n^2}{n} \\ &= \frac{(1+n)^{m\tau\lambda} - 1 \mod n^2}{n} \\ &= \frac{nm\tau\lambda \mod n^2}{n} = \frac{nm}{n} \\ &= m \end{aligned}$$

Correctness of Variant 3. The variant takes $x = m, y = r$ in Problem 2. It is easy to check that

$$\begin{aligned} \frac{c^\lambda - 1 \mod n^2}{n} &= \frac{(g^m r^n)^\lambda - 1 \mod n^2}{n} \\ &= \frac{(g^\lambda)^m - 1 \mod n^2}{n} \\ &= \frac{(1+n)^m - 1 \mod n^2}{n} \\ &= m \end{aligned}$$

Correctness of Variant 4. It is easy to see that

$$\begin{aligned} &\frac{\frac{c}{(c^d \mod n)^e} - 1 \mod n^2}{n} \\ = &\frac{\frac{(1+n)^m r^e}{(((1+n)^m r^e)^d \mod n)^e} - 1 \mod n^2}{n} \\ = &\frac{\frac{(1+n)^m r^e}{r^e} - 1 \mod n^2}{n} \\ = &\; m \end{aligned}$$

5.2 The Bresson-Catalano-Pointcheval Encryption Scheme Revisited

The Bresson-Catalano-Pointcheval encryption scheme has not directly specified that $n \,\vert\, \mathrm{ord}_{n^2}(g)$. But it is easy to find that such a picked g satisfies the condition with high probability. In view of that the condition is necessary to recover x in Problem 1 (see the proof of Theorem 1), we shall directly specify it in the Setup phase.

The random pad r is chosen by the sender and is blinded as

$$A = g^r \mod n^2, \quad B = h^r(1+mn) \mod n^2.$$

We have

$$A = g^r \cdot 1^n \mod n^2,$$

it here takes $x = r, y = 1$ in Problem 1. Thus one knowing the trapdoor λ can recover r using Paillier's computational method. Note that although B could be viewed as

$$B = (1+n)^m h^r \mod n^2.$$

It does not fall into the class of Problem 1. One cannot recover r from B whether the trapdoor is known or not.

After r is retrieved, one can recover $m = \frac{B/h^r - 1 \bmod n^2}{n}$ directly. Obviously, the original computational method incurs more cost.

Based on the observation, we now present a new revision of the scheme (see Table 5).

We stress that the new alternative decryption method does not invoke the secret parameter a, which means the secret parameters a, λ, ρ can be divided into two groups, $\{a\}$ and $\{\lambda, \rho\}$. The two groups of secret parameters can be allocated to different users so as to enhance the robustness of key management. The new version is more flexible to be integrated into other cryptographic schemes.

Table 4: Some variants of Paillier's encryption scheme

	$n = pq$ is an RSA modulus, $\lambda = \text{lcm}(p-1, q-1)$.
Variant 1	$g \in \mathbb{Z}_{n^2}^*$, $\text{ord}_{n^2}(g) = \alpha n$. PK: n, g; SK: α.
(Paillier)	$m \in \mathbb{Z}_n$, $r \in \mathbb{Z}_n$, $c = g^{m+rn} \bmod n^2$.
	$m = \left(\frac{c^\alpha - 1 \bmod n^2}{n}\right) / \left(\frac{g^\alpha - 1 \bmod n^2}{n}\right) \bmod n$
Variant 2	$\kappa = \tau\lambda$, $\tau = \lambda^{-1} \bmod n$. PK: n; SK: κ.
(Damgård-Jurik)	$m \in \mathbb{Z}_n$, $r \in \mathbb{Z}_n$, $c = (1+mn)r^n \bmod n^2$.
	$m = \frac{c^\kappa - 1 \bmod n^2}{n}$
Variant 3	$g^\lambda = 1 + n \bmod n^2$. PK: n, g; SK: λ.
(Choi-Choi-Won)	$m \in \mathbb{Z}_n$, $r \in \mathbb{Z}_n$, $c = g^m r^n \bmod n^2$.
	$m = \frac{c^\lambda - 1 \bmod n^2}{n}$
Variant 4	$e < n$, $d = e^{-1} \bmod \phi(n)$. PK: n, e; SK: d.
(Catalano-Gennaro	$m \in \mathbb{Z}_n$, $r \in \mathbb{Z}_n$, $c = (1+mn)r^e \bmod n^2$.
-Howgrave-Nguyen)	$m = \frac{\frac{c}{(c^d \bmod n)^e} - 1 \bmod n^2}{n}$

Table 5: The Bresson-Catalano-Pointcheval encryption scheme revisited

	The original	The revisited
Setup	$n = pq$, $\lambda = \text{lcm}(p-1, q-1)$. $\alpha \in \mathbb{Z}_{n^2}^*$, $a < n\lambda/2$, $g = \alpha^2 \bmod n^2, h = g^a \bmod n^2$. $\rho = \left(\frac{g^\lambda - 1 \bmod n^2}{n}\right)^{-1} \bmod n$. $\tau = \lambda^{-1} \bmod n$. PK: n, g, h; SK: a, λ, ρ, τ.	$n = pq$, $\lambda = \text{lcm}(p-1, q-1)$. $g \in \mathbb{Z}_{n^2}^*$, $n \mid \text{ord}_{n^2}(g)$. $a \in \mathbb{Z}_n^*$, $h = g^a \bmod n^2$. $\rho = \left(\frac{g^\lambda - 1 \bmod n^2}{n}\right)^{-1} \bmod n$. PK: n, g, h; SK: a, λ, ρ.
Enc.	For $m \in \mathbb{Z}_n$, pick $r \in \mathbb{Z}_n$, compute $A = g^r \bmod n^2$, $B = h^r(1+mn) \bmod n^2$. The ciphertext is $c = (A, B)$.	It is the same as the original.
Dec. 1	$m = \frac{B/A^a - 1 \bmod n^2}{n}$	It is the same as the original.
Dec. 2	$r = \rho\left(\frac{A^\lambda - 1 \bmod n^2}{n}\right) \bmod n$. $\gamma = ar \bmod n$. $m = \frac{\left(\frac{B}{g^\gamma}\right)^\lambda - 1 \bmod n^2}{n} \cdot \tau \bmod n$,	$r = \rho\left(\frac{A^\lambda - 1 \bmod n^2}{n}\right) \bmod n$. $m = \frac{B/h^r - 1 \bmod n^2}{n}$

Table 6: Comparisons of Paillier's encryption and some variants

The original	$c = g^m r^n \bmod n^2$. $n \mid \mathrm{ord}_{n^2}(g), x = m, y = r$. Verification w.r.t. (m, s_1, s_2): $H(m) \overset{?}{=} g^{s_1} s_2^n \bmod n^2$. **True**.
Variant 1	$c = g^m (g^r)^n = g^{m+rn} \bmod n^2$. $\mathrm{ord}_{n^2}(g) = \alpha n, x = m,\ y = g^r$ is a special random pad. Verification w.r.t. (m, s_1, s_2): $H(m) \overset{?}{=} g^{s_1+s_2 n} \bmod n^2$. **False**.
Variant 2	$c = (1+n)^m r^n = (1+mn) r^n \bmod n^2$. $g = 1+n, \mathrm{ord}_{n^2}(g) = n, x = m, y = r$ Verification w.r.t. (m, s_1, s_2): $H(m) \overset{?}{=} (1+s_1 n) s_2^n \bmod n^2$. **False**.
Variant 3	$c = g^m r^n \bmod n^2$. $g^\lambda = 1+n, x = m, y = r$ It can **not** resist a chosen ciphertext attack.
Variant 4	$c = (1+mn) r^e \bmod n^2$, $g = 1+n, ed = 1 \bmod \phi(n)$. Verification w.r.t. (m, s_1, s_2): $H(m) \overset{?}{=} (1+s_1 n) s_2^e \bmod n^2$. **False**.
Variant 5 (Bresson-Catalano -Pointcheval)	$(A, B) = (g^r \bmod n^2, (1+mn) h^r \bmod n^2)$ Verification w.r.t. (m, s_1, s_2): $H(m) \overset{?}{=} (1+s_1 n) h^{s_2} \bmod n^2$. **False**.

5.3 Comparisons

The Paillier's encryption scheme can be naturally converted into a signature scheme because it can retrieve the random pad r as well as the message m. This is due to that it only requires $n \mid \mathrm{ord}_{n^2}(g)$. But in the Variant 1, one cannot retrieve $r \in \mathbb{Z}_n^*$, instead $g^r \bmod n^2$. Though the Variant 3 is very similar to the original Paillier's encryption scheme, it is insecure against a chosen ciphertext attack [22]. The others, Variant 2, Variant 4 and Variant 5 cannot be converted into signature schemes. See Table 6 for details.

By the way, the claim that some variants are more efficient than the original Paillier's encryption scheme is somewhat misleading. Actually, in Paillier's encryption scheme the computation

$$\left(\frac{g^\lambda - 1 \bmod n^2}{n}\right)^{-1} \bmod n$$

has no relation to the ciphertext c. It can be computed and stored previously. The dominated computation in the decryption procedure is that

$$\frac{c^\lambda - 1 \bmod n^2}{n},$$

while the corresponding computation in Variant 4 is

$$m = \frac{\frac{c}{(c^d \bmod n)^e} - 1 \bmod n^2}{n},$$

and that in Variant 5 is

$$m = \frac{B/A^a - 1 \bmod n^2}{n}.$$

We find these decryptions require almost the same computational cost.

6 Conclusion

We revisit the Paillier's cryptosystem and present an efficient alternative decryption procedure for Bresson-Catalano-Pointcheval encryption scheme.

We reaffirm that the original Paillier's encryption scheme has a special property that it naturally implies a signature scheme, while those variants miss this feature.

We would like to stress that although a homomorphic encryption allows anyone to perform some computations on encrypted data, despite not having the secret decryption key, the computations are constrained to the underlying domain (finite fields or rings). A misapplication of a homomorphic encryption for numerical calculations can give rise to errors.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Project 61303200, 61411146001). The authors gratefully acknowledge the reviewers for their valuable suggestions.

References

[1] D. Boneh, C. Craigentry, S. Halevi, F. Wang, and D. Wu, "Private database queries using somewhat homomorphic encryption," in *Proceedings of Applied Cryptography and Network Security (ACNS'13)*, pp. 102–118, Banff, AB, Canada, June 2013.

[2] E. Bresson, D. Catalano, and D. Pointcheval, "A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications," in *Proceedings of Advances in Cryptology (ASIACRYPT'03)*, pp. 37–54, Banff, AB, Canada, Nov. 2003.

[3] Z. J. Cao and L. H. Liu, "Comment on 'harnessing the cloud for securely outsourcing large-scale systems of linear equations'," *IEEE Transactions on Parallel Distributed Systems*, vol. 27, pp. DOI 10.1109/TPDS.2016.2531669, 2016.

[4] G. Castagnos and F. Laguillaumie, "Linearly homomorphic encryption from ddh," in *Proceedings of the Cryptographer's Track at the RSA Conference (CT-RSA'15)*, pp. 487–505, San Francisco, CA, USA, 2015.

[5] D. Catalano, A. Marcedone, and O. Puglisi, "Authenticating computation on groups: New homomorphic primitives and applications," in *Proceedings of Advances in Cryptology (ASIACRYPT'14)*, pp. 193–212, Kaoshiung, Taiwan, Dec. 2014.

[6] T. Y. Chen, C. C. Lee, M. S. Hwang, and J. K. Jan, "Towards secure and efficient user authentication scheme using smart card for multi-server environments," *Journal of Supercomputing*, vol. 66, no. 2, pp. 1008–1032, 2013.

[7] D. Choi, S. Choi, and D. Won, "Improvement of probabilistic public key cryptosystem using discrete logarithm," in *Proceedings of Information Security and Cryptology (ICISC'01)*, pp. 72–80, Banff, AB, Canada, Dec. 2001.

[8] D. Choi, S. Choi, and D. Won, "Paillier's cryptosystem revisited," in *Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS'01)*, pp. 206–214, Philadelphia, Pennsylvania, USA, Nov. 2001.

[9] I. Damgård and J. Jurik, "A generalisation, a simplification and some applications of paillier's probabilistic public-key system," in *Proceedings of Public Key Cryptography (PKC'01)*, pp. 119–136, Cheju Island, Korea, Feb. 2001.

[10] I. Damgård, J. Jurik, and J. Nielsen, "A generalization of paillier's public-key system with applications to electronic voting," *International Journal of Information Security*, no. 9, pp. 371–385, 2010.

[11] D. Galbraith, "Elliptic curve paillier schemes," *Journal of Cryptology*, vol. 15, no. 2, pp. 129–138, 2002.

[12] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC'09)*, pp. 169–178, Bethesda, MD, USA, May 2009.

[13] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Compter and System Sciences*, vol. 28, no. 2, pp. 270–299, 1984.

[14] W. F. Hsien, C. C. Yang, and M. S. Hwang, "A survey of public auditing for secure data storage in cloud computing," *International Journal of Network Security*, vol. 18, no. 1, pp. 133–142, 2016.

[15] M. Joye and B. Libert, "Efficient cryptosystems from 2k-th power residue symbols," in *Proceedings of Advances in Cryptology (EUROCRYPT'13)*, pp. 76–92, Athens, Greece, May 2013.

[16] C. C. Lee, M. S. Hwang, and S. F. Tzeng, "A new convertible authenticated encryption scheme based on the elgamal cryptosystem," *International Journal of Foundamation Computer Science*, vol. 20, no. 2, pp. 351–359, 2009.

[17] C. W. Liu, W. F. Hsien, C. C. Yang, and M. S. Hwang, "A survey of public auditing for shared data storage with user revocation in cloud computing," *International Journal of Network Security*, vol. 18, no. 4, pp. 650–666, 2016.

[18] C.W. Liu, W. F. Hsien, C. C. Yang, and M. S. Hwang, "A survey of attribute-based access control with user revocation in cloud data storage," *International Journal of Network Security*, vol. 18, no. 5, pp. 900–916, 2016.

[19] D. Naccache and J. Stern, "A new public key cryptosystem based on higher residues," in *Proceedings of the 5th ACM Conference on Computer and Communications Security (CCS'98)*, pp. 546–560, San Francisco, CA, USA, Nov. 1998.

[20] T. Okamoto and S. Uchiyama, "A new public-key cryptosystem as secure as factoring," in *Proceedings of Advances in Cryptology (EUROCRYPT'98)*, pp. 308–318, Espoo, Finland, May 1998.

[21] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of Advances in Cryptology (EUROCRYPT'99)*, pp. 223–238, Prague, Czech Republic, May 1999.

[22] K. Sakurai and T. Takagi, "On the security of a modified paillier public-key primitive," in *Proceedings of 7th Australian Conference on Information Security and Privacy (ACISP'02)*, pp. 436–448, Melbourne, Australia, July 2002.

[23] B. Schoenmakers and P. Tuyls, "Efficient binary conversion for paillier encrypted values," in *Proceedings of Advances in Cryptology (EUROCRYPT'06)*, pp. 522–537, St. Petersburg, Russia, May 2006.

[24] C. Wang, K. Ren, J. Wang, and Q. Wang, "Harnessing the cloud for securely outsourcing large-scale systems of linear equations," *IEEE Transaction on Parallel Distributed Systems*, vol. 24, no. 6, pp. 1172–1181, 2013.

Zhengjun Cao is an associate professor with the Department of Mathematics at Shanghai University. He received his Ph.D. degree in applied mathematics from Academy of Mathematics and Systems Science, Chinese Academy of Sciences. He served as a post-doctor in

Computer Sciences Department, Université Libre de Bruxelles, from 2008 to 2010. His research interests include cryptography, discrete logarithms and quantum computation.

Lihua Liu is an associate professor with the Department of Mathematics at Shanghai Maritime University. She received her Ph.D. degree in applied mathematics from Shanghai Jiao Tong University. Her research interests include combinatorics, cryptography and information security.

An Improvement of Fermat's Factorization by Considering the Last m Digits of Modulus to Decrease Computation Time

Kritsanapong Somsuk, Kitt Tientanopajai
(Corresponding author: Kritsanapong Somsuk)

Department of Computer Engineering, Faculty of Engineering, Khon Kaen University
Khon Kaen 40002, Thailand
(Email: tonpor007@hotmail.com)

(Received Dec. 9, 2015; revised and accepted Jan. 23, 2016)

Abstract

Fermat's Factorization Algorithm (FFA) and the algorithms improved from FFA are the fast integer factorization algorithms when these algorithms are chosen to find two large prime factors of the balanced modulus. The key is a process to find two perfect squares such that their difference is equal to the modulus. However, it is time-consuming to find these two integers because there is only one solution but many integers are chosen in this experiment to find the solution. In this paper, a new improvement of FFA is proposed by leaving out some unrelated integers, which do not affect getting the correct solution. Leaving out these integers results from analyzing the last m digits of the modulus where m is a positive integer. The new faster and improved algorithm is called Specific Fermat's Factorization Algorithm Considered from X (SFFA-X) where X is represented as the last m digits of the modulus. The experimental results showed that SFFA-X can factor the modulus faster than FFA and many modified algorithms of FFA especially when at least 2 digits of X are chosen for the implementation.

Keywords: Fermat's factorization algorithm, integer factorization, RSA

1 Introduction

Integer Factorization is one of the famous techniques for breaking RSA [10] which is the most well-known public key cryptosystem. In general, if the modulus is factored as prime numbers, the private key kept secret will be recovered and then RSA is broken [7]. At present, many integer factorization algorithms were proposed such as [1, 2, 3, 4, 6, 8, 9, 11, 15, 18, 19]. However, the speed of each factorization algorithm for factoring the modulus depends on the size of the modulus and the size of prime factors of the modulus. Nevertheless, Fermat's Factorization Algorithm (FFA) [1, 18] discovered by Pierre de Fermat, is the efficient factorization algorithm whenever it is used to factor the balanced modulus that the difference between two large prime factors is very small [18]. In addition, to find the two large prime factors of the modulus, FFA will rewrite the modulus as the difference of the perfect squares. Although, many factorization algorithms improved from FFA [6, 13, 14, 16, 17, 18] were introduced, they are still time-consuming to factor the modulus.

Assume n is represented as the modulus which is equal to the product of two prime numbers and all prime numbers can be chosen to be a prime factor of n except 2 and 5. In this paper, an efficient technique to speed up FFA is proposed. This technique will consider the last m digits of n before choosing one of the new proposed specific algorithms for the implementation. The advantage of the specific algorithms is the removing unrelated iterations of the computation. In addition, the specific algorithms for the last $m + 1$ digits of n can factor n faster than the specific algorithms for the last m digits of n because the details of n are better known and more iterations of the computation are left.

In fact, the numbers of the specific algorithms for the last m digits of n are based on values of m. The numbers of these algorithms are $4 * 10^{m-1}$. For example, if m is equal to 1, there are 4 specific algorithms as follows: the algorithm for the last digit of n is 1, 3, 7 or 9. Another example assumes m is equal to 2 then there are 40 specific algorithms as follows: the algorithm for the last 2 digits of n is 01, 03, 07, 09, 11, $\cdots$, 97 or 99. The new fast and proposed algorithm is called Specific Fermat's Factorization Algorithm Considered from X (SFFA-X), where X is represented as the last m digits of n. For example, the specific algorithm for the last 2 digits of $n = 1287901$ is called SFFA-01, the specific algorithm for the last 2 digits of $n = 737$ is called SFFA-37 and the specific algorithm for last 3 digits of $n = 136313$ is called SFFA-313.

2 Priliminary

2.1 Fermat's Factorization Algorithm: FFA

FFA is one of some integer factorization algorithms which can factor all composite integers. The speed of FFA depends on the difference between two large prime factors of n: $n = p * q$, where p and q are prime numbers and p is larger than q, and the size of n. However, this algorithm is appropriate with the small result of the subtraction between p and q. The key of FFA is that n is rewritten as the difference of perfect squares as follows:

$$n = (\frac{p+q}{2})^2 - (\frac{p-q}{2})^2 \tag{1}$$

In fact, FFA can be distinguished as 2 different algorithms as follows. For the first algorithm, is called FFA-1, assign $x = \frac{p+q}{2}$ and $y = \frac{p-q}{2}$ then,

$$n = x^2 - y^2. \tag{2}$$

However, assign the initial value of x is equal to $\lceil\sqrt{n}\rceil$, then the process to factor n by using FFA-1 is to find the integer of y, $y = \sqrt{x^2 - n}$. If the integer of y is found, the values of p and q can be computed from $p = x + y$ and $q = x - y$. On the other hand, the value of x must be increased by 1 when y is not an integer to compute the new value of y.

Algorithm 1 FFA-1

```
1: Begin
2: Initialize the value of $x = \lceil\sqrt{n}\rceil$
3: $y = \sqrt{x^2 - n}$
4: while $y$ is not an integer do
5:     $x = x + 1$
6:     $y^2 = x^2 - n$
7:     $y = \sqrt{y^2}$
8: end while
9: $p = x + y$
10: $q = x - y$
11: End
```

In Algorithm 1, it implies that the total iterations for computing x and y are same; their total iterations are $(p+q)/2 - \lceil\sqrt{n}\rceil$. However, FFA-1 must take time to compute the square root of integer for all iterations. At present, many modified factorization algorithms modified from FFA-1 were proposed to decrease computation time. The key of some algorithms which will be mentioned in Section 2.2 to 2.5 is to remove some of unrelated steps in the main loop.

The process of the second algorithm, is called FFA-2, is different from FFA-1. This algorithm does not compute the square root of integer as follows: From Equation (1), we have

$$4n = u^2 - v^2$$

where $u = p+q$ and $v = p-q$, therefore, the aim of FFA-2 is to find the corrected values of u and v, respectively.

However, the initial values of u and v are $2\lceil\sqrt{n}\rceil$ and 0, respectively. Assign $r = u^2 - v^2 - 4n$, then two prime factors of n, $p = \frac{u+v}{2}$ and $q = \frac{u-v}{2}$, are found whenever the value of r which is equal to 0 is found. However, two conditions of r are considered when r is not equal to 0 as follows:

Condition 1 ($r > 0$)**:**
The value of v is too small but the value of r is too large. Therefore, v must be increased. On the other hand r must be decreased:

$$\begin{aligned} r &= r - (4v + 4) \\ v &= v + 2. \end{aligned} \tag{3}$$

From Equation (3), the value of $4v + 4$ is from: $(v+2)^2 - v^2 = 4v + 4$. Because, $v = p - q$ is always an even number, the value of v can be increased by 2.

Condition 2 ($r < 0$)**:**
The values of u and r are too small. Therefore, they have to be increased:

$$\begin{aligned} r &= r + (4u + 4) \\ u &= u + 2 \end{aligned} \tag{4}$$

From Equation (4), the value of $4u + 4$ is from: $(u+2)^2 - u^2 = 4u + 4$ Because, $u = p + q$ is always an even number like the value of v, the value of u can be increased by 2. Therefore, the algorithm of FFA-2 for finding two prime factors of n, p and q, is as follows.

Algorithm 2 FFA-2

```
1: Begin
2: Initialize the value of $u = 2\lceil\sqrt{n}\rceil$ and $v = 0$
3: $r = u^2 - v^2 - 4n$
4: while $r$ is not equal to zero do
5:     if $r$ is more than zero then
6:         $r = r - (4v + 4)$
7:         $v = v + 2$
8:     else
9:         $r = r + (4u + 4)$
10:        $u = u + 2$
11:    end if
12: end while
13: $p = \frac{u+v}{2}$
14: $q = \frac{u-v}{2}$
15: End
```

In Algorithm 2, it implies that the value of u is always increased by 2 whenever the value of r is less than 0. On the other hand, the value of v is always increased by 2 whenever the value of r is more than 0. In general, Algorithm 2 shows that the total iterations for computing u and v are different, total iterations of u and v are $(p+q) - 2\lceil\sqrt{n}\rceil$ and $p - q$ in order.

2.2 Modified Fermat Factorization Version 2: MFFV2

MFFV2 [14] does not compute the value of y, Step 6 (or Line 7) of FFA-1, whenever the least significant digit of y^2 is 2, 3, 7 or 8 because y is not certainly an integer. That means the computation time is decreased, although the iterations in the loop are still stable.

2.3 Modified Fermat Factorization Version 3: MFFV3

In 2014, MFFV3 [16] was proposed by using Difference's Least Significant Digit Table (DLSDT) for removing some values of y^2 and y, Step 5 - 6 of FFA-1, from the computation. DLSDT will show the result of the least significant digit of y^2 without the computation of y^2 directly. That means it does not take time to compute y^2 and y when the least significant digit of y^2 is equal to 2, 3, 7 or 8. Furthermore, the information in DLSDT implies that the sequence of 10 iterations in the loop of FFA-1, MFFV3 can remove the 4 steps for computing y^2 and y when the least significant digit of n is 1 or 9. However, this algorithm becomes removing the 6 steps to compute y^2 and y when the least significant digit of n is 3 or 7.

2.4 Modified Fermat Factorization Version 4: MFFV4

MFFV4 [13] can remove more steps for computing y^2 and y when compared with MFFV3. This algorithm uses the new table called Y2MOD20 for analyzing the result of y^2 modulo 20 without computing y^2 directly. From the mathematical theorem, the result of perfect square modulo 20 is always equal to 0, 1, 4, 5, 9 or 16 [9]. Therefore, if the result of y^2 modulo 20 is not equal to 0, 1, 4, 5, 9 or 16, then y^2 is not certainly a perfect square. However, for MFFV4, the value of n is divided into 8 cases. Each case is from the result of n modulo 20 which is 1, 3, 7, 9, 11, 13, 17 or 19. In addition, the sequence of 10 iterations in the loop of FFA-1, MFFV4 can remove 7 steps to compute y^2 and y when the result of n modulo 20 is 1, 9, 11 or 19. Nevertheless, this algorithm can remove 8 steps when the result of n modulo 20 is 3, 7, 13 or 17. Furthermore, the experiment showed that MFFV4 is the fastest improved Fermat's factorization algorithm when compared to FFA-1, MFFV2 and MFFV3.

2.5 Possible Prime Modified Fermat Factorization: P^2MFF

The concept of P^{2}MFF [17] is different from all improved Fermat's factorization algorithms which had been mentioned. In general, this algorithm will be divided into 2 sub algorithms. Each algorithm is arised from the result of n modulo 6. If the result is equal to 5, the form of n is $6k$ - 1 and the form of x must be always $3k_1$, k and k_1 are any integers. That means all iterations of the computation are from the result of $x\%3 = 0$. On the other hand, if the result of n modulo 6 is equal to 1, the form of n is $6k+1$ and the form of x must not be $3k_1$. That means all iterations of the computation are from the result of x%3 = 1 or 2.

3 The Proposed Method

Notation: Assume $z, z_1, z_2 \in Z^+$ and $z_1 \geq z_2$

1) $LSG_1(z)$ = LSG(z) is the last digit of z
2) LSG($z_1 + z_2$) = LSG($LSG(z_1)$ + LSG(z_2))
3) LSG($z_1 - z_2$) = LSG(LSG(z_1) $-$ LSG(z_2)+10)
4) LSG($z_1 * z_2$) = LSG(LSG(z_1) $*$ LSG(z_2))
5) $LSG_m(z)$ is the last m digits of $z, m > 1$
6) $LSG_m(z_1 + z_2) = LSG_m(LSG_m(z_1) + LSG_m(z_2))$
7) $LSG_m(z_1 - z_2) = LSG_m(LSG_m(z_1) - LSG_m(z_2) + 10^m)$
8) $LSG_m(z_1 * z_2) = LSG_m(LSG_m(z_1) * LSG_m(z_2))$

For the notation above, if the result of $LSG(z_1) - LSG(z_2)$ or $LSG_m(z_1) - LSG_m(z_2)$ is a negative integer, it will be increased by 10 or 10^m respectively for changing the result to be the positive integer.

Example 1.

$$\begin{aligned}
LSG(31) &= 1\\
LSG(17+21) &= LSG(LSG(17)+LSG(21))\\
&= LSG(7+1) = LSG(8) = 8\\
LSG(38-15) &= LSG(LSG(38)-LSG(15)+10)\\
&= LSG(8-5+10) = LSG(13) = 3\\
LSG(32-15) &= LSG(LSG(32)-LSG(15)+10)\\
&= LSG(2-5+10) = LSG(7) = 7\\
LSG(5*14) &= LSG(LSG(5)*LSG(14))\\
&= LSG(5*4) = LSG(20) = 0.
\end{aligned}$$

The objective of this paper is to propose the new modified algorithm improved from FFA-2. The key is that the algorithm of FFA-2 will be divided into many sub algorithms based on the last m digits of n. These algorithms are called Specific Fermat's Factorization Algorithm Considered from X (SFFA-X) where X is represented as the last m digits of n. SFFA-X can leave out some values of u and v which are not the expected values. Nevertheless, if $LSG_m(n)$ is considered, numbers of SFFA-X for last m digits of n are based on the values of m that is equal to $4*10^{m-1}$.

In fact, the set of the possible values of u and v for SFFA-X are from the following definition: (Assign a, b, p_t and $q_t \in Z^+$ and n is represented as the modulus).

$$\begin{aligned}
S \;=\; &\{(a, b, p_t, q_t, n) \mid a = LSG_m(p_t),\\
&b = LSG_m(q_t) \;\text{ and }\; LSG_m(ab) = LSG_m(n)\}
\end{aligned}$$

From the set of S, if the last m digits of the multiplication result between the last m digits of p_t, $a = LSG_m(p_t)$, and the last m digits of q_t, $b= LSG_m(q_t)$, is equal to the last m digits of n, $LSG_m(n) = LSG_m(ab)$, then, the values of $LSG_m(u)$ and $LSG_m(v)$ can be computed from $LSG_m(a + b)$ and $LSG_m(a - b)$ or $LSG_m(b - a)$, respectively.

In general, all possible values of $LSG_m(u)$ and $LSG_m(v)$ can be found by considering the both following theorems.

Theorem 1. *Assign a and b are any positive odd integers, where $LSG_m(a) = a_m a_{m-1} \cdots a_2 a_1$, $LSG_m(b) = b_m b_{m-1} \cdots b_2 b_1$ and $m > 1$, that the result of $LSG_m(ab)$ is equal to $LSG_m(n)$. If a_1 is equal to b_1, then the results of $LSG_m((a + 10^{m-1})(b + 9(10^{m-1})))$ and $LSG_m((a + 9(10^{m-1}))(b + 10^{m-1}))$ are also equal to $LSG_m(n)$.*

Proof. Assign: $LSG_m(ab) = c_m c_{m-1} \cdots c_2 c_1$, $LSG_m(a + b) = LSG_m(u)$, $LSG_m(a - b) = LSG_m(v_1)$, $LSG_m(b - a) = LSG_m(v_2)$ and $c_i = d_i\%10$, where $i = 1, 2, \cdots , m$.

From the multiplication technique, we have

$$\begin{aligned} d_1 &= a_1 b_1 \\ d_2 &= a_2 b_1 + a_1 b_2 + \lfloor \frac{d_1}{10} \rfloor \\ d_m &= a_m b_1 + a_{m-1} b_2 + \cdots + a_1 b_m + \lfloor \frac{d_{m-1}}{10} \rfloor \end{aligned}$$

Case 1: Assign $LSG_m(a + 10^{m-1}) = e_m e_{m-1} \cdots e_2 e_1$, $LSG_m(b+9(10^{m-1})) = f_m f_{m-1} \cdots f_2 f_1$, $LSG_m((a+10^{m-1})(b + 9(10^{m-1}))) = g_m g_{m-1} \cdots g_2 g_1$ and $g_i = h_i\%10$. Because $LSG_{(m-1)}(a) = LSG_{(m-1)}(a + 10^{m-1})$ and $LSG_{(m-1)}(b) = LSG_{(m-1)}(b + 9(10^{m-1}))$, hence $e_j = a_j$, $f_j = b_j$ and $h_j = d_j$ where $j = 1, 2, \cdots , m - 2, m - 1$. In addition, $e_m = (a_m+1)\%10$ and $f_m = (b_m+9)\%10$. Therefore, From the multiplication technique, we have

$$\begin{aligned} g_m &= (e_m f_1 + e_{m-1} f_2 + \cdots + e_1 f_m + \lfloor \frac{h_{m-1}}{10} \rfloor)\%10 \\ &= ((a_m + 1)b_1 + a_{m-1} b_2 + \cdots + a_1(b_m + 9) \\ &\qquad +\lfloor \frac{d_{m-1}}{10} \rfloor)\%10 \\ &= (a_m b_1 + a_{m-1} b_2 + \cdots + a_1 b_m + \lfloor \frac{d_{m-1}}{10} \rfloor \\ &\qquad +(b_1 + 9a_1))\%10. \end{aligned}$$

Because $a_1 = b_1$, that means $(b_1 + 9a_1)\%10 = 0$. Hence, $g_m = (a_m b_1 + a_{m-1} b_2 + \cdots + a_1 b_m + \lfloor \frac{d_{m-1}}{10} \rfloor)\%10 = c_m$.

Moreover, g_j is also equal to c_j because a_j and b_j are always equal to e_j and f_j, respectively. Therefore, $LSG_m((a+10^{m-1})(b+9(10^{m-1})))$ is always equal to $LSG_m(ab)$ that means it is also equal to $LSG_m(n)$.

However, this case implies that for any pair of a and b that a_1 and b_1 are fixed, there is only one value of $LSG_m(u)$, $LSG_m((a + 10^{m-1}) + (b + 9(10^{m-1}))) = LSG_m(a + b + 10^m) = LSG_m(a + b) = LSG_m(u)$. On the other hand, the value of $LSG_m(v_1)$ is always decreased by $8(10)^{m-1}$, $LSG_m((a + 10^{m-1}) - (b + 9(10^{m-1}))) = LSG_m(v_1 - 8(10)^{m-1})$, and the value of $LSG_m(v_2)$ is always increased by $8(10)^{m-1}$, $LSG_m((b + 9(10^{m-1})) - (a + 10^{m-1})) = LSG_m(v_2 + 8(10^{m-1}))$, where v_1 and v_2 are represented as the possible values of v.

Case 2: Assign $LSG_m(a + 9(10^{m-1}))=e_m e_{m-1} \cdots e_2 e_1$, $LSG_m(b + 10^{m-1})= f_m f_{m-1} \cdots f_2 f_1$, $LSG_m((a + 9(10^{m-1}))(b + 10^{m-1})) = g_m g_{m-1} \cdots g_2 g_1$ and $g_i = h_i\%10$.

Because $LSG_{(m-1)}(a) = LSG_{(m-1)}(a + 9(10^{m-1}))$ and $LSG_{(m-1)}(b) = LSG_{(m-1)}(b + 10^{m-1})$, hence $e_j = a_j, f_j = b_j$ and $h_j = d_j$ where $j = 1, 2, \cdots , m - 2, m - 1$. In addition, $e_m = (a_m + 9)\%10$ and $f_m = (b_m + 1)\%10$.

From the multiplication technique, we have

$$\begin{aligned} g_m &= (e_m f_1 + e_{m-1} f_2 + \cdots + e_1 f_m + \lfloor \frac{h_{m-1}}{10} \rfloor)\%10 \\ &= ((a_m + 9)b_1 + a_{m-1} b_2 + \cdots + a_1(b_m + 1) \\ &\qquad +\lfloor \frac{d_{m-1}}{10} \rfloor)\%10 \\ &= (a_m b_1 + a_{m-1} b_2 + \cdots + a_1 b_m + \lfloor \frac{d_{m-1}}{10} \rfloor \\ &\qquad +(a_1 + 9b_1))\%10. \end{aligned}$$

Because $a_1 = b_1$, that means $(a_1 + 9b_1)\%10 = 0$. Hence, $g_m = (a_m b_1 + a_{m-1} b_2 + \cdot + a_1 b_m + \lfloor \frac{d_{m-1}}{10} \rfloor)\%10 = c_m$.

Moreover, g_j is also equal to c_j because a_j and b_j are always equal to e_j and f_j, respectively. Therefore, $LSG_m((a+9(10^{m-1}))(b+10^{m-1}))$ is always equal to $LSG_m(ab)$. That means it is also equal to $LSG_m(n)$.

However, this case implies that there is only one value of $LSG_m(u)$, $LSG_m((a+9(10^{m-1})) + (b+10^{m-1}) = LSG_m(a + b) = LSG_m(u)$. On the other hand, the value of $LSG_m(v_1)$ is always increased by $8(10)^{m-1}$, $LSG_m((a+9(10^{m-1})) - (b+10^{m-1})) = LSG_m(v_1 + 8(10)^{m-1})$, and the value of $LSG_m(v_2)$ is always decreased by $8(10)^{m-1}$, $LSG_m((b + 10^{m-1}) - (a + 9(10^{m-1}))) = LSG_m(v_2 - 8(10^{m-1}))$, when v_1 and v_2 are represented as the possible values of v.

□

Theorem 2. *Assign a and b are any positive odd integers, where $LSG_m(a) = a_m a_{m-1} \cdots a_2 a_1$, $LSG_m(b) = b_m b_{m-1} \cdots b_2 b_1$ and $m > 1$, that the result of $LSG_m(ab)$ is equal to $LSG_m(n)$. If a_1 is not equal to b_1 and there are various pairs of positive odd integers, k_1 and k_2, which are equal or smaller than 9 and not equal to 5 that the result of $(a_1 k_2 + b_1 k_1)\%10 = 0$, then the result of $LSG_m((a + 10^{m-1} k_1)(b + 10^{m-1} k_2))$ is also equal to $LSG_m(n)$.*

Proof. Assign: $LSG_m(ab) = c_m c_{m-1} \cdots c_2 c_1$, $LSG_m(a + 10^{m-1}k_1) = x_m x_{m-1} \cdots x_2 x_1$, $LSG_m(b + 10^{m-1}k_2) = y_m y_{m-1} \cdots y_2 y_1$, $LSG_m((a + 10^{m-1}k_1)(b + 10^{m-1}k_2)) = z_m z_{m-1} \cdots z_2 z_1$, $LSG_m(a+b) = LSG_m(u)$, $LSG_m(a-b) = LSG_m(v_1)$, $LSG_m(b-a) = LSG_m(v_2)$, $c_i = d_i\%\ 10$ and $z_i = w_i\%10$, where $i = 1, 2, 3, \cdots, m$.

From the multiplication technique, we have $c_m = (a_m b_1 + a_{m-1} b_2 + \cdots + a_1 b_m + \lfloor \frac{d_{m-1}}{10} \rfloor)\%10$.

Because $LSG_{(m-1)}(a) = LSG_{(m-1)}(a + 10^{m-1}k_1)$ and $LSG_{(m-1)}(b) = LSG_{(m-1)}(b + 10^{m-1}k_2)$, then $x_j = a_j, y_j = b_j$ and $w_j = d_j$ where $j = 1, 2, \cdots, m-2, m-1$. In addition, $x_m = (a_m + k_1)\%10$ and $y_m = (b_m + k_2)$ % 10. Therefore, $z_m = (x_m y_1 + x_{m-1} y_2 + \cdots + x_1 y_m + \lfloor \frac{w_{m-1}}{10} \rfloor)\%10 = ((a_m + k_1)b_1 + a_{m-1}b_2 + \cdots + a_1(b_m + k_2) + \lfloor \frac{d_{m-1}}{10} \rfloor)\%10 = (a_m b_1 + a_{m-1} b_2 + \cdots + a_1 b_m + \lfloor \frac{d_{m-1}}{10} \rfloor + (a_1 k_2 + b_1 k_1))\%10$.

Because the result of $(a_1 k_2 + b_1 k_1)\%10$ is equal to 0, $z_m = (a_m b_1 + a_{m-1} b_2 + \cdots + a_1 b_m + \lfloor \frac{d_{m-1}}{10} \rfloor)\%10 = c_m$.

Moreover, z_j is also equal to c_j because a_j and b_j are always equal to x_j and y_j, respectively. Therefore, $LSG_m((a + 10^{m-1}k_1)(b + 10^{m-1}k_2))$ is always equal to $LSG_m(ab)$ and $LSG_m(n)$ whenever the result of $(a_1 k_2 + b_1 k_1)\%10$ is equal to 0.

However, this theory implies that for any pair of a and b that a_1 and b_1 are fixed, the value of $LSG_m(u)$ is always increased by $10^{m-1}k$, where $k = k_1 + k_2$, as follows:

$$\begin{aligned} & LSG_m((a + 10^{m-1}k_1) + (b + 10^{m-1}k_2)) \\ = \; & LSG_m(a + b + 10^{m-1}k_1 + 10^{m-1}k_2) \\ = \; & LSG_m(a + b + 10^{m-1}(k_1 + k_2)) \\ = \; & LSG_m(a + b + 10^{m-1}k) \\ = \; & LSG_m(u + 10^{m-1}k) \end{aligned}$$

On the other hand, the value of $LSG_m(v_1)$ is always increased by $10^{m-1}l$, where $l = ((k_1 - k_2) + 10)$ % 10 is always a positive integer, as follows:

$$\begin{aligned} & LSG_m((a + 10^{m-1}k_1) - (b + 10^{m-1}k_2)) \\ = \; & LSG_m(a - b + 10^{m-1}k_1 - 10^{m-1}k_2) \\ = \; & LSG_m(a - b + 10^{m-1}(k_1 - k_2)) \\ = \; & LSG_m(a - b + 10^{m-1}(((k_1 - k_2) + 10)\%10)) \\ = \; & LSG_m(a - b + 10^{m-1}l) \\ = \; & LSG_m(v_1 + 10^{m-1}l). \end{aligned}$$

And the value of $LSG_m(v_2)$ is always increased by $10^{m-1}h$, where $h = ((k_2 - k_1) + 10)$ % 10 is always a positive integer, as follows:

$$\begin{aligned} & LSG_m((b + 10^{m-1}k_2) - (a + 10^{m-1}k_1)) \\ = \; & LSG_m(b - a + 10^{m-1}k_2 - 10^{m-1}k_1) \\ = \; & LSG_m(b - a + 10^{m-1}(k_2 - k_1)) \\ = \; & LSG_m(b - a + 10^{m-1}(((k_2 - k_1) + 10)\%10)) \\ = \; & LSG_m(b - a + 10^{m-1}h) \\ = \; & LSG_m(v_2 + 10^{m-1}h). \end{aligned}$$

Therefore, we can conclude that the values of $LSG_m(u)$, $LSG_m(v_1)$ and $LSG_m(v_2)$ are always increased by $10^{m-1}k$, $10^{m-1}l$ and $10^{m-1}h$, respectively, where k, l and h are any positive integers and v_1 and v_2 are represented as the possible values of v. □

Notation: If k_1 and k_2 are an even number or equal to 5, some values of $LSG_m(u)$ and $LSG_m(v)$ cannot be computed because we cannot find some pairs of $LSG_m(a)$ and $LSG_m(b)$ that $LSG_m(ab) = LSG_m(n)$.

From both of two theorems above, assume a and b are represented as any positive odd integers which the last digit is not equal to 5 and all pairs of $LSG_{(m-1)}(a)$ and $LSG_{(m-1)}(b)$ that $LSG_{(m-1)}(ab)$ is equal to $LSG_{(m-1)}(n)$ are known. All values of $LSG_m(u)$ and $LSG_m(v)$ will be found whenever only one pair of $LSG_m(a)$ and $LSG_m(b)$ that $LSG_m(ab)$ is equal to $LSG_m(n)$ for each pair of $LSG_{(m-1)}(a)$ and $LSG_{(m-1)}(b)$ is found. In deep, all the other pairs of $LSG_m(a)$ and $LSG_m(b)$ are computed by using Theorem 1 when $LSG(a)$ is equal to $LSG(b)$ or using Theorem 2 when $LSG(a)$ is not equal to $LSG(b)$, until the repeated solution is found.

Due to all pairs of $LSG(a)$ and $LSG(b)$ that $LSG(ab)$ is equal to $LSG(n)$ cannot be computed by using Theorem 1 and Theorem 2, therefore these pairs must be considered directly. Table 1 shows all possible pairs of $LSG(a)$ and $LSG(b)$ for all possible values of $LSG(n)$.

Generally, Table 1 shows the following information:

1) If $LSG(n)$ equals to 1, there are 3 possible pairs of ($LSG(a)$, $LSG(b)$), 3 values of $LSG(u)$ and 3 values of $LSG(v)$.

2) If $LSG(n)$ equals to 3, there are 2 possible pairs of ($LSG(a)$, $LSG(b)$), 2 values of $LSG(u)$ and 2 values of $LSG(v)$.

3) If $LSG(n)$ equals to 7, there are 2 possible pairs of ($LSG(a)$, $LSG(b)$), 2 values of $LSG(u)$ and 2 values of $LSG(v)$.

4) If $LSG(n)$ equals to 9, there are 3 possible pairs of ($LSG(a)$, $LSG(b)$), 3 values of $LSG(u)$ and 3 values of $LSG(v)$.

Assume all pairs of ($LSG_{(m-1)}(a)$, $LSG_{(m-1)}(b)$) that the last $m-1$ digits of their multiplication which is equal to $LSG_{(m-1)}(n)$ are known. To complete the speed up for factoring n by improving FFA-2, the algorithm of SFFA-X is divided into 3 algorithms.

First is the main algorithm. This algorithm is used to find two prime factors, p and q.

Moreover, it implies that some unrelated values of U and V which their last m digits are not in the sets of $LSG_m(u)$ and $LSG_m(v)$ are left out from the computation. Leaving out values of U and V is from

Table 1: All possible values of LSG(u), LSG(v) and pairs of LSG(a) and LSG(b) when considered from LSG(n)

LSG(n)	**Pair of (LSG(a),LSG(b))**	**LSG(LSG(a)+LSG(b)) (LSG(u))**	**LSG(LSG(a)-LSG(b)) or LSG(LSG(b)-LSG(a)) (LSG(v))**
1	(1, 1)	2	0
	(3, 7)	0	6 or 4
	(9, 9)	8	0
3	(1, 3)	4	8 or 2
	(7, 9)	6	8 or 2
7	(1, 7)	8	4 or 6
	(3, 9)	2	4 or 6
9	(1, 9)	0	2 or 8
	(3, 3)	6	0
	(7, 7)	4	0

Algorithm 3 FFA-X

```
1: Begin
2: Compute all members of $LSG_m(u)$, $LSG_m(v)$, $disu$ and $disv$ by using Algorithm 4. In deep, $disu$ is the set of the subtraction results between two adjacent values of $LSG_m(u)$ and $disv$ is the set of the subtraction results between two adjacent values of $LSG_m(v)$.
3: Find the initial values of $U$ and $V$ that $LSG_m(U)$ and $LSG_m(V)$ must equal to one of all possible values of $LSG_m(u)$ and $LSG_m(v)$, in order. In addition, $U$ and $V$ are started at $2\lceil\sqrt{n}\rceil$ and 0 respectively.
4: Assign $i$ is the index of $disu$ that the initial value is based on the values of $LSG_m(U)$ and the position in $LSG_m(u)$ and $disu$.
5: Assign $j$ which is always started at 0 is the index of $disv$.
6: $r = U^2 - V^2 - 4n$
7: while $r$ is not equal to zero do
8:     if $r$ is more than zero then
9:         $r = r - (2 * V * disv(j) + disv(j)^2)$
10:        $V = V + disv(j)$
11:        $j = j + 1$ // $j$ becomes to zero whenever $j$ is equal to the size of $disv$
12:    else
13:        $r = r + (2 * U * disu(i) + disu(i)^2)$
14:        $U = U + disu(i)$
15:        $i = i+1$, // $i$ becomes to zero whenever $i$ is equal to the size of $disu$
16:    end if
17: end while
18: $p = \frac{U+V}{2}$
19: $q = \frac{U-V}{2}$
20: End
```

choosing only the pairs of ($LSG_m(a)$, $LSG_m(b)$) which $LSG_m(LSG_m(a)$*$LSG_m(b)) = LSG_m(ab)$ is equal to $LSG_m(n)$.

Second is the algorithm for finding all members of $LSG_m(u)$, $LSG_m(v)$, $disu$ and $disv$ before returning these variables to Step 1 of Algorithm 3.

Algorithm 4 Finding $LSG_m(u)$, $LSG_m(v)$, $disu$ and $disv$

```
1: Begin
2: Assign each pair of ($LSG_{(m-1)}(a)$, $LSG_{(m-1)}(b)$) is a group.
3: For each group, find a pair of ($LSG_m(a)$, $LSG_m(b)$) using Algorithm 5 and then leave out a pair of ($LSG_{(m-1)}(a)$, $LSG_{(m-1)}(b)$) from the group.
4: For each group, find all pairs of ($LSG_m(a)$, $LSG_m(b)$) by using theorem 1 whenever $LSG(a)$ is equal to $LSG(b)$. However, theorem 2 is used for the other case.
5: Compute all possible values of $LSG_m(u)$ and $LSG_m(v)$. However, all repeated values of $LSG_m(u)$ and $LSG_m(v)$ will be left out from the sets.
6: Sort the members of $LSG_m(u)$ and $LSG_m(v)$ from the minimum to the maximum.
7: Find $disu$, the set of the subtraction results between two adjacent values of $LSG_m(u)$. Nevertheless, the last member of $disu$ is the subtraction between the minimum and the maximum of $LSG_m(u)$. In addition, the result of the last member must be also increased by $10^m$ for changing as the positive integer.
8: Find $disv$, the set of the subtraction results between two adjacent values of $LSG_m(v)$. Nevertheless, the last member of $disv$ is the subtraction between the minimum and the maximum of $LSG_m(v)$. Moreover, this result must be increased by $10^m$.
9: End
```

However, the results from this algorithm can be applied with all values of $LSG_m(n)$ which $LSG_m(u)$, $LSG_m(v)$, $disu$ and $disv$ had been computed. Therefore, only the first time of the computation will be computed to find all pairs of ($LSG_m(a)$, $LSG_m(b)$) to compute all members of these four variables.

Furthermore, if $disu$ or $disv$ have the repeated patterns, we can leave out them from the sets.

Example 2. *Assume, all members of $LSG_2(u)$ and $LSG_2(v)$ of SFFA-03 are known. Finding disu and disv* *of SFFA-03:*

1) *$LSG_2(u)$ = {04, 16, 24, 36, 44, 56, 64, 76, 84, 96}. Therefore, disu = {12, 8, 12, 8, 12, 8, 12, 8, 12, 8}. Because, disu has the repeated patterns, then, we can reassign disu = {12, 8}.*

2) *$LSG_2(v)$ = {02, 18, 22, 38, 42, 58, 62, 78, 82, 98}. Therefore, disv = {16, 4, 16, 4, 16, 4, 16, 4, 16, 4}. Because, disv has the repeated patterns, then, we can reassign disv = {16, 4}.*

In deep, if the value of all members of disu (or disv) is same, these members can be reduced to be only one member of the set. For example, if disv ={20, 20, 20, 20}, we can reduce as disv = 20 and j will be left out from the algorithm because the value of all members in disv is same.

The last algorithm is for computing the pair of ($LSG_m(a)$, $LSG_m(b)$) when a pair of ($LSG_{(m-1)}(a)$, $LSG_{(m-1)}(b)$) is known. The idea of this algorithm is to find only the value of $LSG_m(b)$ while $LSG_m(a)$ is always equal to $0LSG_{(m-1)}(a)$. For example, assume $LSG_3(a) = 841$, then $LSG_4(a) = 0841$.

Algorithm 5 Finding Pair of $LSG_m(a)$ and $LSG_m(b)$

```
1: Begin
2: $x = LSG_{(m-1)}(a)*LSG_{(m-1)}(b)$
3: $n_m = \lfloor \frac{n\%10^m}{10^{m-1}} \rfloor$
4: $x_m = \lfloor \frac{x\%10^m}{10^{m-1}} \rfloor$
5: $a_1 = \text{LSG}_{(m-1)}(a)\ \%\ 10$
6: Assign $i = 0$
7: $j = (i * a_1 + x_m)\%10$
8: while $j$ is not equal to $n_m$ do
9:     $i = i + 1$
10:    $j = (i * a_1 + x_m)\%10$
11: end while
12: $LSG_m(a) = 0LSG_{(m-1)}(a)$
13: $LSG_m(b) = LSG_{(m-1)}(b) + i * 10^{m-1}$
14: End
```

Example 3. *Assign $LSG_3(a) = 233$ and $LSG_3(b) = 897$, $LSG_3(ab) = 001$. Find a pair of ($LSG_4(a), LSG_4(b)$) that $LSG_4(ab) = 2001$ by using Algorithm 5.*

In this example, $LSG_4(ab) = 2001$ can be represented as the value of $LSG_4(n)$.

$$\begin{aligned} x &= 233 * 897 = 209,001 \\ n_4 &= \lfloor \frac{2001\%10^4}{10^3} \rfloor = 2 \\ x_4 &= \lfloor \frac{209,001\%10^4}{10^3} \rfloor = 9 \\ a_1 &= 233\%10 = 3 \end{aligned}$$

where $i = 0, j = (0 * 3 + 9)\%10 = 9, j$ is not equal to 2. $i = 1, j = (1 * 3 + 9)\%10 = 2, j$ is equal to 2.

Then, $LSG_4(a) = 0233$ and $LSG_4(b) = LSG_3(b) + i * 10^3 = 897 + 1000 = 1897$. However, for some values of n, the implementation of SFFA-X should be divided into 2 groups, based on the relation between LSG(a) and LSG(b). In fact, there are 2 cases of SFFA-X as follows:

Case 1: *The algorithm is always divided into 2 groups whenever there may be the pairs of a and b that LSG(a) is equal to LSG(b).*

Group 1: *The algorithm for all pairs of a and b that LSG(a) is equal to LSG(b), the value of LSG(v) is always equal to 0.*

Group 2: *The algorithm for all pairs of a and b that LSG(a) is not equal to LSG(b), the value of LSG(u) is always equal to 0.*

Case 2: *There is only one group whenever there is no pairs of a and b that LSG(a) is equal to LSG(b), both of LSG(u) and LSG(v) are not always equal to 0.*

For the case of n that must be divided into 2 groups, there is only one solution in only one group. Therefore, the process is ended when the corrected solution in one out of two groups is found.

Example 4. *Factoring $n = 1287901$ using SFFA-X with $m = 2$.*
This example is assigned to use SFFA-X with m = 2 to find two prime factors of n. Because $LSG_2(n)$ = 01, therefore SFFA-01 is the algorithm used for the implementation. However, all steps to find two prime factors of n = 1287901 using SFFA-01 are as follows:
Assumption: *All pairs of (LSG(a), LSG(b)) that LSG(ab) is equal to LSG(n) = 1 are known.*

Algorithm 3:

Step 1: *Find all of $LSG_2(u)$, $LSG_2(v)$, disu and disv by using Algorithm 4.*
Algorithm 4:

Step 1: *Group 1: (1, 1); Group 2: (3, 7); Group 3: (9, 9).*

Step 2: *Group 1: (01, 01); Group 2: (03, 67); Group 3: (09, 89). This process is computed by using Algorithm 5.*

Step 3:

Group 1: *(01, 01), (11, 91), (21, 81), (31, 71), (41, 61), (51, 51);*

Group 2: *(03, 67), (13, 77), (23, 87), (33, 97), (43, 07), (53, 17), (63, 27), (73, 37), (83, 47), (93, 57). In this case, one of all possible pairs of (k_1, k_2) which are (1, 1), (3, 3), (7, 7) or (9, 9) is chosen.*

Group 3: *(09, 89), (19, 79), (29, 69), (39, 59), (49, 49), (99, 99).*

Steps 4 - 5: *Because there are some pairs of a and b that LSG(a) is equal to LSG(b), SFFA-01 must be divided into 2 group. Group 1 is from the combining between Group 1 and Group 3, LSG(a) is equal to LSG(b). The other is that LSG(a) is not equal to LSG(b). Therefore,*

Group 1: $LSG_2(u) = \{02, 98\}$; $LSG_2(v) = \{00, 20, 40, 60, 80\}$.

Group 2: $LSG_2(u) = \{10, 30, 50, 70, 90\}$. $LSG_2(v) = \{36, 64\}$.

Steps 6-7 *Group 1:* $disu = \{96, 4\}$ *and* $disv = \{20, 20, 20, 20, 20\} = 20$, *do not assign j. Group 2:* $disu = \{20, 20, 20, 20, 20\} = 20$, *do not assign i, and* $disv = \{28, 72\}$.

End of Algorithm 4.

Steps 2-4: *Find the initial value of U, V, i and j for each group. First, compute* $U = 2\lceil\sqrt{n}\rceil = 2270$.

Group 1: *Because* $LSG_2(U) = 70$ *is not a member in the set of* $LSG_2(u)$, *U must be changed as 2298, the nearest value which is more than 2270. That means the initial value of i must be equal to 1. However, the initial value of V is 0 because the minimum value of* $LSG_2(v)$ *is 0.*

Group 2: *Because* $LSG_2(U) = 70$ *is already a member in the set of* $LSG_2(u)$, $U = 2270$ *can be used as the initial value. However, the variable, i, is not used in this group. In addition, the initial value of V is 36 because the minimum value of* $LSG_2(v)$ *is 36.*

Step 5:

Group 1: $r = 2298^2 - 0^2 - 4(1287901) = 129200$.

Group 2: $r = 2270^2 - 36^2 - 4(1287901) = \boldsymbol{0}$.

In Step 5, the expected solution is found in Group 2, r = 0. Therefore, the process in the loop, Steps 6 - 16, will not be implemented. However, the two prime factors can be computed from $p = \frac{U+V}{2} = \frac{2270+36}{2} = 1153$ *and* $p = \frac{U-V}{2} = \frac{2270-36}{2} = 1117$.

Moreover, for the value of n in Example 3, assume SFFA-901, m = 3, is chosen to factor n instead of using SFFA-01. All pairs of $(LSG_2(a), LSG_2(b))$ *which are found in this example will be used in Step 1 of SFFA-901 in order to find all pairs of* $(LSG_3(a), LSG_3(b))$ *that* $LSG_3(LSG_3(a) * LSG_3(b))$ *is equal to* $LSG_3(n) = 901$.

Example 5. *Factoring n = 133901 using SFFA-X with m = 2.*
This example is assigned $LSG_2(n) = 01$. *That means SFFA-01 is chosen for this example. Because* $LSG_2(u)$, $LSG_2(v)$, *disu and disv of SFFA-01 had been already computed in Example 4, it is not time-consuming to calculate them again and we can start the process at Step 2 of Algorithm 3.*

Algorithm 3:

Steps 2 - 4: *Find the initial value of U, V, i and j for each group. First, compute* $U = 2\lceil\sqrt{n}\rceil = 732$.

Group 1: *Because* $LSG_2(U) = 32$ *is not a member in the set of* $LSG_2(u)$, *U must be changed as 798. That means the initial value of i must be equal to 1. However, the initial value of V is 0 because the minimum value of* $LSG_2(v)$ *is 0.*

Group 2: *Because* $LSG_2(U) = 32$ *is a not member in the set of* $LSG_2(u)$, *U must be changed as 750. However, the variable, i, is not used in this group. In addition, the initial value of V must be 36 because the minimum value of* $LSG_2(v)$ *is 36.*

Step 5:

Group 1: $r = 798^2 - 0^2 - 4(133901) = 101200$.

Group 2: $r = 750^2 - 36^2 - 4(133901) = 25600$.

Steps 6 - 16: *Process in loop:*

Iteration 1:

Group 1: $(r > 0, V = 0)$:

$$\begin{aligned} r &= r - (2 * V * disv + disv^2) = 100800 \\ V &= V + disv = 20. \end{aligned}$$

Group 2: $(r > 0, V = 36, j = 0)$:

$$\begin{aligned} r &= r - (2 * V * disv(0) + disv(0)^2) = 22800. \\ V &= V + disv(0) = 64, j = 1. \end{aligned}$$

Iteration 2:

Group 1: $(r > 0, V = 20)$:

$$\begin{aligned} r &= r - (2 * V * disv + disv^2) = 99600. \\ V &= V + disv = 40. \end{aligned}$$

Group 2: $(r > 0, V = 64, j = 1)$:

$$\begin{aligned} r &= r - (2 * V * disv(1) + disv(1)^2) = 8400. \\ V &= V + disv(1) = 136, j = 0. \end{aligned}$$

Iteration 3:

Group 1: $(r > 0, V = 40)$:

$$\begin{aligned} r &= r - (2 * V * disv + disv^2) = 97600. \\ V &= V + disv = 60. \end{aligned}$$

Group 2: $(r > 0, V = 136, j = 0)$:

$$\begin{aligned} r &= r - (2 * V * disv(0) + disv(0)^2) = \boldsymbol{0}. \\ V &= V + disv(0) = 164. \end{aligned}$$

Because the expected value of r which is equal to 0 is found in Group 2 of the 3^{rd} *iteration, the process is stopped and then two prime factors can be computed from* $p = \frac{U+V}{2} = \frac{750+164}{2} = 457$ *and* $q = \frac{U-V}{2} = \frac{750-164}{2} = 293$.

4 Results and Discussion

In this work, the computer specifications for the implementation are Intel(R) Core(TM) i3 CPU M380 2.53 GHz, 4.00 GB RAM Memory, and Microsoft Windows 8.1 Pro Operating System. Java Programming Language is chosen to develop the algorithms. Moreover, we use BigInteger class which is the class of Java as the data type because this class can be represented as the unlimited data type. The experiment is distinguished as 4 parts. Each bits size in each experiment is the average result of 50 values of n chosen randomly. In addition, for Figure 1 and Figure 2, only the difference of bits size between two prime factors equal to 2 is chosen. However, all algorithms of SFFA-X in these experiments are started at Step 2 of Algorithm 3 because the process of Step 1 is always implemented only the first time of SFFA-X, when X is fixed.

The experiment in Figure 3 is for the same size of two prime factors of n and the values of n that SFFA-X must be divided into 2 groups, $LSG_4(n) = 0001$ is the representative of this experiment. That means only 4 algorithms of SFFA-X in which X is equal to 1, 01, 001 and 0001 are chosen for the implementation. The experiment shows that SFFA-0001 is the fastest algorithm. Nevertheless, the average computation time of FFA-2, SFFA-1, SFFA-01, SFFA-001, SFFA-0001, MFFV4 and P^{2}MFF are about 30.27, 17.17, 4.09, 2.05, 1.82, 5.78 and 15.15 seconds respectively. Furthermore, the information in this figure implies that SFFA-X begins to factor n faster than MFFV4 when X which is equal to 01 is chosen.

The difference between the experiment in Figure 3 and Figure 1 is that the size of two large prime factors in Figure 1 is different while the other in Figure 3 is same. Therefore, all algorithms of SFFA-X for this experiment are still SFFA-1, SFFA-01, SFFA-001 and SFFA-0001. However, the experiment in Figure 1 shows that SFFA-0001 is still the fastest integer factorization algorithm. Nevertheless, the average computation time of FFA-2, SFFA-1, SFFA-01, SFFA-001, SFFA-0001, MFFV4 and P^{2}MFF are about 243.2, 135.76, 29.4, 19.02, 10.73, 227.4 and 341.9 seconds respectively. Furthermore, this figure implies that SFFA-X begins to factor n faster than MFFV4 when X is equal to 1 is chosen.

Whereas, SFFA-X in Figure 4 will not be divided into 2 groups because all values of $LSG_4(n)$ in this experiment is 0003. However, the size of two prime factors of n is same. That means only 4 algorithms of SFFA-X in which X is equal to 3, 03, 003 and 0003 are chosen to implement. The experiment shows that SFFA-0003 is the fastest algorithm. Nevertheless, the average computation time of FFA-2, SFFA-3, SFFA-03, SFFA-003, SFFA-0003, MFFV4 and P^{2}MFF are about 26.4, 9.35, 3.38, 3.2, 1.16, 4.78 and 13.08 seconds respectively. Furthermore, the information in this figure implies that SFFA-X begins to factor n faster than MFFV4 when X which is equal to 03 is chosen.

The experiment in Figure 2 is similar to the other in Figure 1 but LSG$_4(n)$ in Figure 2 is equal to 0003. Therefore, all algorithms of SFFA-X for this experiment are SFFA-3, SFFA-03, SFFA-003 and SFFA-0003. The experimental in Figure 2 shows that SFFA-0003 is the fastest integer factorization algorithm. Nevertheless, the average computation time of FFA-2, SFFA-3, SFFA-03, SFFA-003, SFFA-0003, MFFV4 and P^{2}MFF are about 270.17, 109.7, 49.38, 43.56, 26.5, 167.25 and 448.28 seconds respectively. Furthermore, this figure implies that SFFA-X begins to factor n faster than MFFV4 when X which is equal to 3 is chosen.

Moreover, if we consider the information in Figure 3 and Figure 4, the average computation time of SFFA-0001 and SFFA-0003 are faster than MFFV4 by about 68.56% and 75.75%, respectively. On the other hand, if the information in Figure 1 and Figure 2 is considered, the average computation time of SFFA-0001 and SFFA-0003 becomes faster than MFFV4 by about 95.28% and 84.15%, respectively.

These results imply that SFFA-X is the better choice to factor n when compared with MFFV4 especially when bits size of X is large and the size of two large prime factors is different. The reason is as follows.

Due to MFFV4 and SFFA-X are modified chronologically from FFA-1 and FFA-2, we will compare the iterations of the computation between FFA-1 and FFA-2 instead of their improvements.

The total iterations in the loop of u, in FFA-2, are $(p+q) - 2\lceil\sqrt{n}\rceil$. However, the increment value of u is always 2, but the increment value of x, in FFA-1, is always 1. That means the total iterations of u are equal to the total iterations of x which are equal to $(p+q)/2 - \lceil\sqrt{n}\rceil$. Nevertheless, the total iterations in the loop of v, in FFA-2, are $p-q$ and are more than the total iterations in the loop of y, in FFA-1, the reason are as follows: Assume A and B are represented as the total iterations in the loop of y and v, respectively. Then,

$$\begin{aligned} A &= (p+q) - 2\lceil\sqrt{n}\rceil \\ &\approx (p+q) - 2\sqrt{n} \\ &= (p+q) - 2\sqrt{p}\sqrt{q} \\ &= (\sqrt{p}-\sqrt{q})^2 \\ &= (\sqrt{p}-\sqrt{q})(\sqrt{p}-\sqrt{q}) \qquad (5) \\ B &= p-q \\ &= \sqrt{p}^2 - \sqrt{q}^2 \\ &= (\sqrt{p}-\sqrt{q})(\sqrt{p}+\sqrt{q}) \qquad (6) \end{aligned}$$

From Equations (5) and (6), all possible results are divided into 2 conditions:

Condition 1: (p is close to q)
It is obvious that the iterations of FFA-2 are greater than FFA-1. Therefore, the condition of SFFA-X which is faster than MFFV4 is that the digits of X in this case must be large enough, the digits of X in Figure 3 and Figure 4 must be at least 2.

Condition 2: (p is far from q)

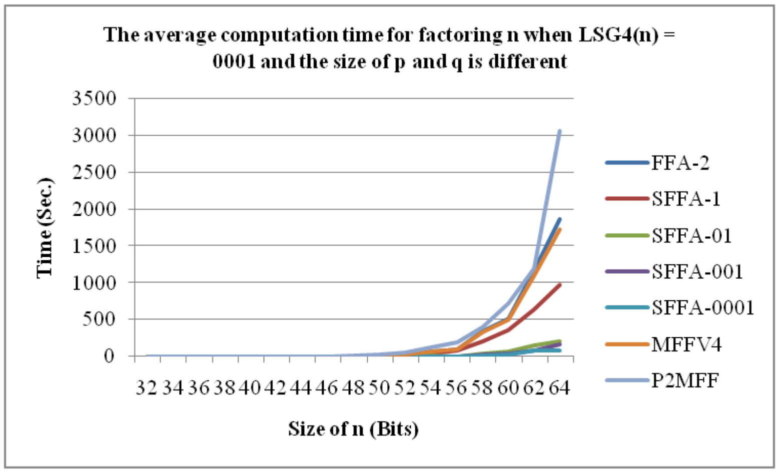


Figure 1: Average computation time for factoring $LSG_4(n) = 0001$ and the size of p and q is different

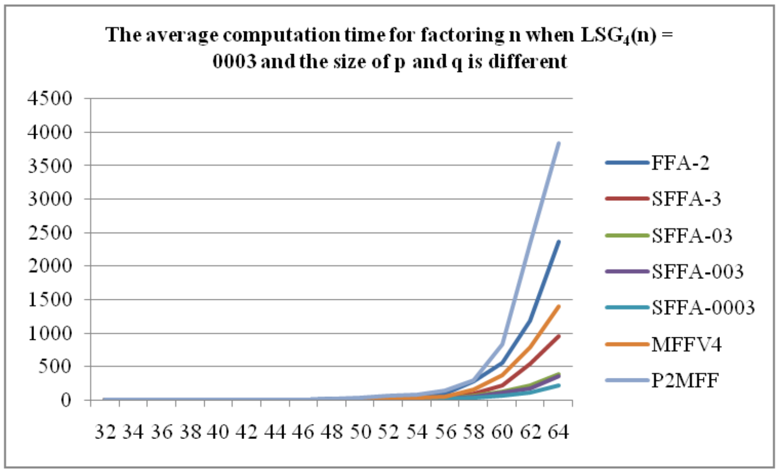


Figure 2: Average computation time for factoring $LSG_4(n) = 0003$ and the size of p and q is different

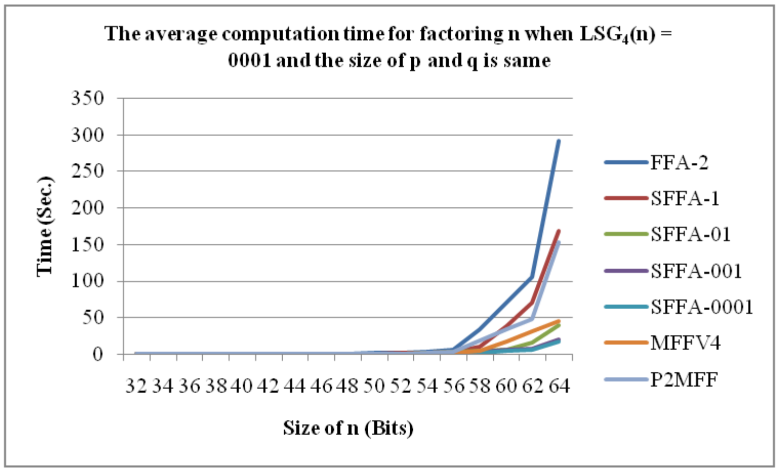


Figure 3: Average computation time for factoring $LSG_4(n) = 0001$ and the size of p and q is same

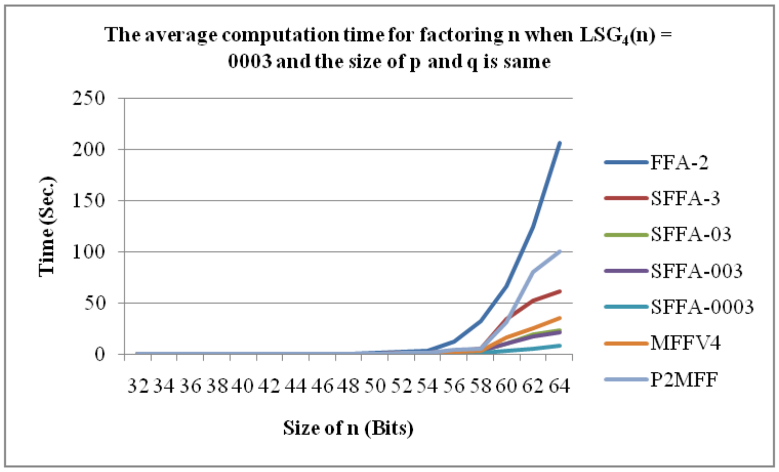


Figure 4: Average computation time for factoring $LSG_4(n) = 0003$ and the size of p and q is same

If p is very larger than q, then q may be left out from the equation to estimate total iterations. That means the iterations of FFA-1 are close to FFA-2. However, FFA-1 is time-consuming to compute the square root of integer while FFA-2 does not to do. This reason indicates that FFA-2 is certainly faster than FFA-1. That means most of SFFA-X can factor n faster than MFFV4 although the size of X is smaller than the other in Condition 1, the digit of X in Figure 1 and Figure 2 is only 1.

From both of two conditions above, we concluded that most of SFFA-X can factor n faster than MFFV4 especially when the size of X is large enough.

Moreover, total iterations of SFFA-X can be found from the following equation:

$$\begin{aligned} t &= (j * \lfloor \frac{A - u_x}{\sum_{i=0}^{j-1} disu(i)} \rfloor + c_u) \\ &\quad +(k * \lfloor \frac{B - LSG_m(v_0)}{\sum_{i=0}^{k-1} disv(i)} \rfloor + c_v). \end{aligned}$$

Where,

1) t = total iterations of SFFA-X in main loop;

2) $A = (p + q) - 2\lceil\sqrt{n}\rceil$;

3) $B = p - q$;

4) u_x = the increment value of $2\lceil\sqrt{n}\rceil$ to get the initial value of U;

5) j = size of $disu$;

6) k = size of $disv$;

7) $LSG_m(v_0)$ = the minimum value of $LSG_m(v)$;

8) c_u is the remainder iterations that are more than 1 but less than size of $disu$ when there is the remainder of $\frac{A-u_x}{\sum_{i=0}^{j-1} disu(i)}$. However, c_u is equal to 0 when there is not the remainder;

9) c_v is the remainder iterations that are more than 1 but less than size of $disv$ when there is the remainder of $\frac{B-LSG_m(v_0)}{\sum_{i=0}^{k-1} disv(i)}$. However, c_v is equal to 0 when there is not the remainder.

However c_u and c_v can be found by using Algorithm 6 and Algorithm 7 respectively.

Algorithm 6 Calculating c_u

```
1: Begin
2: $size\_disu$ = size of $disu$ that the repeated patterns are removed
3: $count\_u$ = the index of $LSG_m(u)$ that is equal to the initial value of $LSG_m(U)$
4: $count\_u = count\_u \% size\_disu$
5: $s\_u = \sum_{i=0}^{j-1} disu(i)$
6: $r\_u = (A - u_x)\% s\_u$
7: while $r\_u$ is not equal to zero do
8:     $r\_u = r\_u - disu(count\_u)$
9:     $count\_u = count\_u + 1$
10:    if $count\_u$ is equal to $size\_disu$ then
11:        $count\_u = 0$
12:    end if
13: end while
14: $c_u = count\_u$
15: End
```

In addition, for some value of n that SFFA-X must be divided into 2 group, t is computed by using only all parameters in the solution group.

Example 6. *Find total iterations in Example 5.*

Because the solution group is in Group 2, we have to use parameters in this group as follows:

1) $A = (457 + 293) - 732 = 18$*;*

2) $B = 457 - 293 = 164$*;*

3) $j = 1$*, size of* $disu$*;*

Algorithm 7 Calculating c_v

```
1: Begin
2: $count\_v = 0$
3: $s\_v = \sum_{i=0}^{k-1} disv(i)$
4: $r\_v = (B - \mathrm{LSG}_m(v_0))\% s\_v$
5: while $r\_v$ is not equal to zero do
6:     $r\_v = r\_v - disv(count\_v)$
7:     $count\_v = count\_v + 1$
8: end while
9: $c_v = count\_v$
10: End
```

4) $\sum_{i=0}^{0} disu(i) = disu = 20$*;*

5) $k = 2$*, size of* $disv$*;*

6) $\sum_{i=0}^{1} disv(i) = disv(0) + disv(1) = 28 + 72 = 100$*;*

7) $u_x = 18$*, the initial value of* U *is 750, then* $U - 2\lceil\sqrt{n}\rceil = 750 - 732 = 18$*;*

8) $LSG_2(v_0) = 36$*;*

9) $\dfrac{A - u_x}{\sum_{i=0}^{j-1} disu(i)} = \dfrac{18-18}{20} = 0$*, the remainder is 0, then* $c_u = 0$*;*

10) $\dfrac{B - LSG_2(v_0)}{\sum_{i=0}^{k-1} disu(i)} = \dfrac{164-36}{100} = 1$*, the remainder is 28 (*$r\_v = 28$*), then* c_v *can be computed by using Algorithm 7 as follows:*

a. $count\_v = 0, s\_v = 100, r\_v = 28$*;*

b. $r\_v = r\_v - disv(0) = 28 - 28 = 0$*;*

c. $count\_v = count\_v + 1 = 1$*.*

Because $r\_v = 0$*, then* $c_v = count\_v = 1$*. Therefore, total iterations in main loop is*

$$t = ((1)(0) + 0) + ((2)(1) + 1) = 3.$$

In general, both of $\frac{j}{\sum_{i=0}^{j-1} disu(i)}$ and $\frac{k}{\sum_{i=0}^{k-1}(i)}$ are always less than $\frac{1}{2}$. Therefore, t is always less than total iterations of FFA-2 that is equal to $\frac{A+B}{2}$. Furthermore, t can be decreased when the value of m is larger because the values of $\frac{j}{\sum_{i=0}^{j-1} disu(i)}$ and $\frac{k}{\sum_{i=0}^{k-1}(i)}$ are certainly smaller.

5 Conclusion

The aim of this paper is to propose a new technique to speed up the 2nd method of Fermat's Factorization Algorithm (FFA-2) by considering the last m digits of n to leave out some values of u and v which are not in the condition. This technique is called Specific Fermat's Factorization Algorithm Considered from X (SFFA-X) where X is the last m digits of n. Furthermore, the concept of this technique implies that the computation time of SFFA-X will be reduced more whenever the bigger size of m is considered because more details of u and v are known. Therefore, the unrelated values of u and v are increased and they should be left out from the computation.

Moreover, SFFA-X can be also applied with Estimated Prime Factor (EPF) [18] using the technique of continued fractions [5, 12] to estimate the new initial values of U and V to reduce more iterations of the computation. However, the applying SFFA-X with EPF can be used to factor only the unbalanced modulus in order to get the high accuracy.

References

[1] A. R. Ambedkar, A. Gupta, P. Gautam, and S. S. Bedi, "An efficient method to factorize the RSA public key encryption," in *Proceedings of Communication Systems and Network Technologies (CSNT'11)*, pp. 108–111, Jummu, Katra, June 2011.

[2] D. Bishop, *Introduction to Cryptography with Java Applets*, Germany: Jones and Bartlett Publisher, 2003.

[3] D. M. Bressoud, *Factorization and Primality Testing*, United states: Springer-Verlag, 2013.

[4] Q. Huang, Y. T. Li, Y. Zhang, and C. Lu, "A modified non-sieving quadratic sieve for factoring simple blur integers," in *Proceedings of Multimedia and Ubiquitous Engineering (MUE'07)*, pp. 729–732, Seoul, South Korea, Apr. 2007.

[5] N. Kobiltz, *A Course in number theory and cryptography*, United states: Springer, 1994.

[6] J. McKee, "Speeding fermat' factoring method," *Mathematics of Computation*, vol. 68, pp. 1729–1738, 1999.

[7] N. A. Moldovyan, "Short signatures from difficulty of factorization problem," *International Journal of Network Security*, vol. 8, no. 1, pp. 90–95, 2009.

[8] J. Pollard, "Monte carlo methods for index computation (mod p)," *Mathematics of Computation*, vol. 32, no. 7, pp. 918–924, 1978.

[9] H. Riesel, *Prime Numbers and Computer Methods for Factorization*, United states: Birkhuser Boston, 1994.

[10] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Communications of ACM*, vol. 21, no. 7, pp. 120–126, 1978.

[11] P. Sharma, A. Gupta, and A. Vijay, "Modified integer factorization algorithm using v-factor method," in *Proceedings of Advanced Computing & Communication Technologies (ACCT'12)*, pp. 423–425, Haryana, India, Jan. 2012.

[12] J. H. Silverman, *A friendly introduction to number theory*, United states: Pearson Education International, 2012.

[13] K. Somsuk, "A new modified integer factorization algorithm using integer modulo 20's technique," in *Proceedings of International Computer Science and Engineering Conference (ICSEC'14)*, pp. 312–316, Khon Kaen, Thailand, July 2014.

[14] K. Somsuk and S. Kasemvilas, "Mffv2 and mnqsv2: Improved factorization algorithms," in *Proceedings of International Conference on Information Science and Applications (ICISA'13)*, pp. 1–3, Pattaya, Thailand, June 2013.

[15] K. Somsuk and S. Kasemvilas, "Mvfactor: A method to decrease processing time for factorization algorithm," in *Proceedings of International Computer Science and Engineering Conference (ICSEC'13)*, pp. 339–342, Bangkok, Thailand, Sept. 2013.

[16] K. Somsuk and S. Kasemvilas, "Mffv3: An improved integer factorization algorithm to increase computation speed," in *Proceedings of The KKU International Conference (KKU-IENC'14)*, pp. 1432–1436, Khon Kaen, Thailand, Mar. 2014.

[17] K. Somsuk and S. Kasemvilas, "Possible prime modified fermat factorization: New improved integer factorization to decrease computation time for breaking rsa," in *Proceedings of International Conference on Computing and Information Technology (IC2IT'14)*, pp. 325–334, Phuket, Thailand, May 2014.

[18] M. E. Wu, R. Tso, and H. M. Sun, "On the improvement of fermat factorization using a continued fraction technique," *Future Generation Computer Systems*, vol. 30, pp. 162–168, 2014.

[19] J. Zalaket and J. H. Boutros, "Prime factorization using square root approximation," *Computers and Mathematics with Applications*, vol. 61, no. 9, pp. 2463–2467, 2011.

Kritsanapong Somsuk is a Ph.D. student of the department of Computer Engineering in Faculty of Engineering, Khon Kaen University, Khon Kaen, Thailand. He received M.Eng. (Computer Engineering) degree from Khon Kaen University in 2009 and M.Sc. (Computer Science) degree from Khon Kaen University in 2013. His research interests are cryptography and integer factorization algorithms.

Kitt Tientanopajai is a lecturer of the department of Computer Engineering in Faculty of Engineering, Khon Kaen University, Khon Kaen, Thailand. He received his M.Eng. (Computer Science and Information Management) degree from Asian Institute of Technology in 1998 and his D.Eng. (Computer Science and Information Management) degree from Asian Institute of Technology in 2005. His research interests include Free/Open Source Software, Information Security, Quality of Service Routing, Computer Networks and Educational Technology.

Using a New Structure in Group Key Management for Pay-TV

Shih-Ming Chen[1], Ching-Rong Yang[1], and Min-Shiang Hwang[1,2]
(Corresponding author: Min-Shiang Hwang)

Department of Computer Science and Information Engineering, Asia University[1]
No. 500, Lioufeng Rd., Wufeng, Taichung 41354, Taiwan
(Email: mshwang@asia.edu.tw)
Department of Medical Research, China Medical University Hospital, China Medical University[2]
No. 91, Hsueh-Shih Road, Taichung 40402, Taiwan
(Received May 17, 2015; revised and accepted Aug. 21 & Sept. 8, 2015)

Abstract

This paper studies a key management problem for a conditional access system with a control word, an authorization key, a distribution key and the master private key. Using a new key store structure and multiple select groups, we can reduce the memory request size and update time. This problem is considered with a large memory for a matrix of group key store structure of four key distributions, and only one group has select right of four key distributions in a free select channel.

Keywords: Conditional access system (CAS), pay-TV, key management, four-key distribution

1 Introduction

Pay-TV is the best business way to distribute a mass of information program to a large number of people simultaneously. Because of already established free payment by users, television has become a widespread communications medium. A Pay-TV system channel provider charges the subscriber fee for receiving the broadcasting program. In real business applications, a Pay-TV system can be a digital broadcasting system (DBS) [3, 14], or a digital cable TV system such as the local cable TV system (CATV) [7, 11]. In an industry society, scheduled programming does not always offer what television receivers desire. Therefore, additional video services provided by CATV network have enjoyed enormous successes during the last decade. A Pay-TV system has many broadcasting channels to provide its subscriber, and these channels can be classified into two classes [15, 20]. The first kind of channel is the basic channel available to all the subscribers of the system. Second kind of channel is the pay-channel that charges the subscriber for the receiving fee. The pay-channel can be classified into two subclasses (See Table 2). The first subclass is Pay Per Channel (PPC) [18], receiving fee for each channel counted according to a time unit. Second subclass is Pay Per View (PPV) [2, 5], counted for each program.

Table 1: The class of broadcasting channels

program	Subscription	PPV	Fee
Basic subscriber	$\times$	$\times$	$\surd$
PPC subscriber	$\surd$	$\times$	$\surd$
PPV subscriber	$\surd$	$\surd$	$\surd$

Where, $\times$: has no viewing right; $\surd$: has viewing right. From Table 1, there exist a lot of charge fee problems on CATV. In technology, registered subscribers could be sufficiently authorized by taking advantage of conditional-access system (CAS) reference upon table [6, 12, 17]. It can follow upon table to permit only the authorized subscribers to watch the CATV program. Thus, CAS constructs a key distribution and management rule for Pay-TV services. The management rules include encryption and decryption keys to refresh those keys periodically and then distribute those keys to authorized subscribers secretly so that unauthorized receivers cannot get the correct keys [13].

This paper is organized as follows: Section 2 discusses a secure message module. Section 3 discusses the basic key for CAS management. Section 4 discusses the proposed key distribution models. Section 5 discusses proposed store structure of the key distribution model and multiple select group police. Section 6 concludes the paper.

2 Secure Message Module

Between the transmitters and receivers, these have two secure message modules multiplexed with the signal itself.

These two modules are introduced as follows:

1) Entitlement Control Module:
   The Entitlement control module is a route Entitlement Control Message (ECM) to microprocessor of the smart card of set-top-box (STB) [1, 8] with the received control parameters. If passing the comparison, authorized receivers can decrypt control word (CW) by an authorized key in the smart card. Thus, the ECM consists of an access parameter and enciphered CW.

2) Entitlement Management Module:
   The Entitlement Management Message (EMM) [19] carries the information of the receiving program to the STB. The Entitlement management includes access rights and updates the AK for the channel subscriber. Mail or a specific channel without the program simultaneously in a batch process can transfer this EMM.

3 Basic Keys

We will describe four elements about CAS management subscriber and key distribution in this section. We discuss their characteristics about control word, authorization key, group key, and master private key for CAS control.

1) Control Word:
   It is used to scramble and descramble broadcasting programs in each charged channel. Each charged channel has a unique control word (CW). Thus, n charge channels have $cw_1 \sim cw_n$. The control word should be updated within a short time period of 5-10 seconds.

2) Authorization Key:
   It is used to encipher the control word and access the ECM. Each charged channel has a unique Authorization key. Thus, n charge channels have $AK_1 \sim AK_n$. The control word should be updated within a day or month period of time.

3) Group Key:
   It is used to encipher the authorization key. Each charge group and receive group have a unique group key [4, 9]. Thus, i charge groups and j receive groups have $RGK_1 \sim RGK_{ij}$. The row of group key should be updated once per month. One row of group key should be updated once per day. Therefore, each row is updated per month. In Tu et al.'s propose, the group key of four-level key management uses a matrix to store [16].

4) Master Private Key:
   It is used to encipher the group key and is unique store in the smart card for each subscriber. Each Master Private Key (MPK) is never charged during the life cycle of the smart card. Thus S subscribers have $MPK_1 \sim MPK_s$.

4 Relate Key Distribution Models

There are many proposed schemes for charge-fee programs of Pay-TV at last. That key management is to use a hierarchy key management method in the proposed scheme. In this section, we will discuss briefly about proposed key distribution models' shortcoming. We assume S subscribers and C channels in Pay-TV system. It will be basic assume in follow methods.

4.1 Two-Key and Three-Key Distributions

The two-key scheme uses an MPK to distribute CW. Thus, The CAS needs to compute CW of every charge channel using only subscribers' MPK. There is $S \times C$ message-packages encrypted and broadcasting within 5 to 20 seconds. In real application, Pay-TV may have hundreds of channels and millions of subscribers. The total broadcasting message package will to be huge load for CAS. The profit of this scheme is simple to implement. But it is not suitable for a large Pay-TV system.

The three-Key distribution scheme [10] is focused on two Key distribution shortcomings. It will reduce a huge load in every update and retransmission. The scheme adds AK between in CW and MPK. The AK is a time variant and unique in every channel. The system encrypts the CW by using the AK. The AK is encrypted by the MPK. The AK is updated every month. Therefore, Total CW only uses S times to encrypt and retransmission message package within 5 to 20 seconds. However, it must refresh AK about $S \times C$ every month. It will be a huge load system in one day of every month. Therefore, the scheme is not suitable for a large system. It is only suitable for a PPV program with a few of channels and few of subscriber.

4.2 Four-key Distribution

We will describe two schemes about Tu et al. [16]. The proposed scheme for CAS management subscribers and key distributions in this section. We will discuss their characteristics and flow about the key transform, encrypted and decrypted for CAS control.

4.2.1 Simple Model

The scheme development is focused on three Key distribution shortcomings. It will reduce a huge load in every update and retransmission. The scheme adds RGK between in AK and MPK. The RGK is a time variant and unique in each receiving channel group. The system encrypts the CW by using the AK. The AK of channels in same receiving channel group is distributed by RGK. The receiving channel group by combining those authorizations keys together is encrypted by RGK of this group. The AK and RGK are updated periodically about one month. The EMM package includes encrypted AK and RGK. The scheme assumes N receiving group for classifications of subscribers. Define the receiving group key matrix as follows.

$$RGK = [rgk_1, rgk_2 \cdots rgk_N]$$

Thus, it must refresh Aks of channels about $S \times C$ to $S + N$ in every month. It will reduce a huge load system in one day each month, so the scheme is not suitable for a dynamic system.

4.2.2 Complete Model

The four-key complete model architecture is used to implement the CAS to provide PPC service for a Pay-TV dynamic key management system. The scheme is focused on a simple model and a charge group for dynamic key management. Therefore, it combines two assuming M charging groups and N receiving groups for the classifications of subscribers. The set S of all subscribers of the system is classified into $M \times N$ classes of the disjoint subscribing class s_{ij} where $1 \leq i \leq M$, $1 \leq j \leq N$:

$$S = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1N} \\ s_{21} & s_{22} & \cdots & s_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ s_{M1} & s_{M2} & \cdots & s_{MN} \end{bmatrix}$$

Each row of S matrix is the same charging groups and each column of S matrix is the same receiving groups. Thus, we can mirror to the receiving group key matrix (RGK) as following:

$$RGK = \begin{bmatrix} rgk_{11} & rgk_{12} & \cdots & rgk_{1N} \\ rgk_{21} & rgk_{22} & \cdots & rgk_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ rgk_{M1} & rgk_{M2} & \cdots & rgk_{MN} \end{bmatrix}$$

The receiving group key is unique for each subscribing class in S matrix. Therefore, we can easily understand rgk_{ij} is the receiving group key of the subscribing class s_{ij}. Each row of RGK matrix is refreshed in a day each month. And each key of column of each row of RGK matrix is encrypted with each subscriber's MPK of s_{ij}. The AKs of PPC channels are updated daily. The AKs of PPV channels are updated per program. Therefore, any subscriber that is out of authorization date will not receive the renewed keys in RGK, so they would not be able to watch programs. If a new subscriber is added to the system, the subscriber must follow the above rule to be classified into a receiving group key class and broadcasted to the new subscriber. If one subscriber wants to move into a new receive class for-vacation, and the CAS wants to delete the subscriber from a receiving class, the receiving group key of this class should be updated and re-broadcast to other subscribers within this class. The moved subscriber without the new receiving group key cannot get the AK and loses the receiving authentication.

5 The Propose Scheme

We propose a new key store structure and multiple select group police for a key distribution scheme from the complete model of four-key distribution. Because the four-key distribution is a good scheme for the key distribution of PPC channels. However, it uses two matrixes of a very large size of memory to store receive and charge a group key and the group of all subscribers. Because the system may have millions of subscribers and hundreds of channels, the potential amount of group to be classified is very large for store receiving group key and the group of all subscriber information in the free selective channel police. These large matrixes have a lot of empty space because subscribers have the same selecting channels or CAS supporter offers channels of discount for a viewing group. We propose a new structure and multiple-select group police to solve a waste of memory space. There are two kinds of structures presented for store key and one new multiple select group police as follows.

5.1 Link-List Structure for a Group Key

In Tu et al.'s proposed scheme is to use two matrices to store a receiving group key and the group of all subscribers. It let matrices have many empty spaces for selecting. First, we propose a new store structure that is the receiving group key Link-List to replace a receiving group key matrix as shown in Figure 1.

$LL_1 \sim LL_M$ charge a group and $N_1 \sim N_M$ receiving group, but $N_1 \sim N_M$ may be not equal. Because it only stores a subscribed group that has an empty space to waste, the scheme refreshes one Link-List daily, and each Link-List is refreshed once per month. The other thing is the same as Tu et al.'s complete model of four-level key distribution. It assumes the CAS has C channels, S subscribers, M charge group and N receiving group. If CAS supports free-select channel police for subscribers, we can easily obtain MAX$M = 31$ and $N = 2^C - 1$ in Tu et al.'s complete model. Thus, In Tu et al.'s matrixes, the complete model of four-key distribution model will request $31 \times 2^C - 1$ memory space for receiving group key matrixes. The system may have millions of subscribers and hundreds of channels, so the potential amount of group to be classified is very large to store a receiving group key and the group of all subscriber information in free-selective channel police. We only needs $2 \times M \times N$ memory in my proposed structure. We can easily obtain $1 \leq M \leq 31$, $1 \leq N \leq 2^C - 1$ for Link-List structure. In the worst, Link-List structure is equal to the matrix for memory request. In Link-List structure scheme, this problem never happens. The scheme can solve the draw-

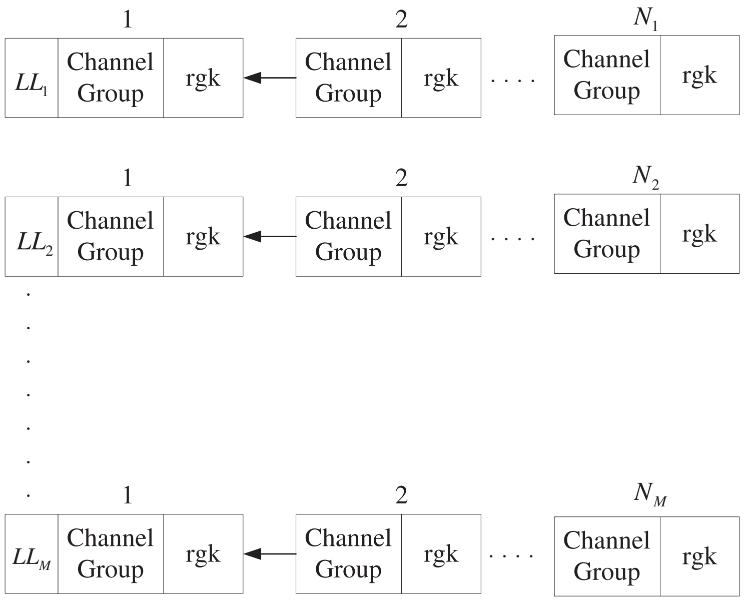


Figure 1: Link-List structure

back of the Tu et al.'s of four-levels key distribution that needs a very large system memory for a group key store matrix.

5.2 Array Structure for Subscribers

The structure is focused on all subscribers of the system for a store matrix. It let a matrix have many empty spaces to waste for subscribers. We propose one array structure to replace the matrix of all subscribers of the system as shown in Table 2. We assume the CAS has N receiver group, M charge group, L Link-List point address memory length and S subscribers. If CAS supports free-select channel police for subscribers, we can easily obtain MAX$M = 31$ and $N = 2^C - 1$ in Tu et al.'s complete model. In Tu et al.'s matrix of the complete model of a four-key distribution model will request $31 \times 2^C - 1$ memory space for the matrix of all subscribers in the system. We only need $S \times L$ memory space in my propose array structure. Thus, we can easily obtain $S \times L$ ¡ $31 \times 2^C - 1$. Let assume CAS have 10000 subscribers, and 4 bytes of memory for Link-List point address and 100 channels. We need $31 \times 2^{100} - 1$-matrix memory for Tu et al's proposed scheme. However, we can reduce memory request from $31 \times 2^{100} - 1$ down to 10000×4 for the array structure.

Table 2: Array structure

Subscriber	Link-List Point Address
⋮	⋮

5.3 Multiple Select Group Police

The scheme is focused on defining select group police for subscribes in all select possibly. In free-select channel police, that will request $31 \times 2^C - 1$ memory requests for the receiving group key of any methods. When this happen, we can choose the multiple select group police to reduce memory request.

The multiple select group police must be deleted including group of many channels that are replaced with some group of minor channel in CAS. However, it lets subscribers can multiple select channel group their want. It means that they can use multiple select to obtain one equal to old one group including many channels. When we use this police, the N receiver group is less than $2^C - 1$ receiver group. If CAS has 4 channels for subscribers to choose, we have 15 kinds of receiver groups for choice. And that will request 31×15 memory spaces for free-select channel police in worst time, if each subscriber selects including 4, 6, 7, 8 channels group to watch TV. However, we can choose two groups, 4, 6 and 7, 8, for multiple select group police. Therefore, receiver group including 4, 6, 7, 8 channels must to be deleted to reduce memory space request. If we can find so many this state and delete this receiver group, it will reduce many memory requests in any methods.

6 Conclusions

Efficient compression and modulation techniques have been implemented for a large-scale Pay-TV broadcasting. It uses a kind of key distribution and management for Pay-TV charge fee. We have discussed above many key distribution models for Pay-TV. We can find three-key distribution model is suitable for PPV programs, and four-key distribution model is suitable for PPC programs for the no-free-select channel application environment.

We have proposed two schemes and one police are Link-List structure, array structure and multiple select police for memory space reduced. In fact, the conditional access control system can allow the Pay-TV for a free-select channel that is subscribers' wish and trend of the time. Thus, we apply the new structure and police for a four-key architecture of CAS to a Pay-TV. Originally, we can use Tu et al.'s four-key distribution architecture to implement, but under the consideration of memory matrix, the Link-List and array structure of four-key architecture are employed. Of course, we can obtain two profit and one shortcoming in my proposed scheme as follow.

Profit:

- The scheme only uses a minimum memory for CAS in Link-List and array structure of a four-key distribution scheme.
- It can select multiple channels in multiple select group police for CAS.

Shortcoming:

- It needs a more complex algorithm than Tu et al.'s complete model of a four-key distribution scheme.

All these issues are emerging; making Tu et al.'s complete model of a four-key distribution scheme concept is good than other proposed schemes. However, Tu et al.'s scheme has two shortcomings that requests a large memory in a complete scheme and lack multiple select groups for a free-select channel in CAS. The scheme we propose can solve these two shortcomings.

Acknowledgements

This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no.: MOST 103-2221-E-468 -026, NSC 103-2622-E-468-001-CC2, and NSC 103-2622-H-468-001-CC2.

References

[1] N. Anwar, I. Riadi, A. Luthfi, "Forensic SIM card cloning using authentication algorithm," *International Journal of Electronics and Information Engineering*, vol. 4, no. 2, pp. 71–81, 2016.

[2] H. W. Chi, G. L. Li, M. J. Chen and J. R. Lin, "Efficient computation allocation algorithm for multiview video coding," *International Journal of Electronics and Information Engineering*, vol. 3, no. 2, pp. 91–106, 2015.

[3] S. J. Crowley, *Capacity Trends in Direct Broadcast Satellite and Cable Television Services*, National Association of Broadcasters, Oct. 8, 2013. (`http://www.nab.org/documents/newsRoom/pdfs/100813_Capacity_Trends_in_DBS_and_Cable_TV_Services.pdf`)

[4] Z. Eslami, M. Noroozi, and S. K. Rad, "Provably secure group key exchange protocol in the presence of dishonest insiders," *International Journal of Network Security*, vol. 18, no. 1, pp. 33–42, 2016.

[5] D. L. Garcia, A. Nebot, A. Vellido, "Visualizing pay-per-view television customers churn using cartograms and flow maps," in *Proceedings of 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'13)*, pp. 567–572, 2013.

[6] D. He, N. Kumar, H. Shen, J. H. Lee, "One-to-many authentication for access control in mobile pay-TV systems," *Science China Information Sciences*, pp. 1–14, Apr. 13, 2016.

[7] D. Hunter, J. Coder, J. Ladbury, "Effects of LTE signals on cable TV devices," in *2014 United States National Committee of URSI National Radio Science Meeting (USNC-URSI NRSM'14)*, 2014.

[8] J. Kim, E. S. Jung, Y. T. Lee, W. Ryu, "Home appliance control framework based on smart TV set-top box," *IEEE Transactions on Consumer Electronics*, vol. 61. no. 3, pp. 279–285, 2015.

[9] A. Kumar and S. Tripathi, "Anonymous ID-based group key agreement protocol without pairing," *International Journal of Network Security*, vol. 18, no. 2, pp. 263–273, 2016.

[10] J. W. Lee, "Key distribution and management for conditional access system on dbs," in *Proceedings of International Conference on Cryptology and Information Security*, pp. 82–88, 1996.

[11] R. Li, N.K. Chung, K.T. MO, D.M. Fisher, and V. Wong, "A flexible display module for dvd and set-top-box applications," *IEEE Transactions on Consumer Electronics*, vol. 43, no. 2, pp. 496–503, 1997.

[12] Y. Liu, X. Yang, H. Yao, and W. Gao, "Novel secure communication protocol for conditional access system," *International Journal of Network Security*, vol. 5, no. 2, pp. 121–127, 2007.

[13] B. M. Macq, J. J. Quisquater, "Cryptology for digital tv broadcasting," *Proceedings of the IEEE*, vol. 83, pp. 944–957, June 1995.

[14] H. Y. Seo, B. Bae, J. D. Kim, "Transmission model for next-generation digital broadcasting systems," in *International Conference on Information Networking (ICOIN'15)*, pp. 379–380, 2015.

[15] C. Y. Sun and C. C. Chang, "Cryptanalysis of a secure and efficient authentication scheme for access control in mobile pay-TV systems," *International Journal of Network Security*, vol. 18, no. 3, pp. 594–596, 2016.

[16] F. K. Tu, C. S. Laih, H. H. Tung, "On key distribution management for conditional access system on pay-tv system," *IEEE Transactions on Consumer Electronics*, vol. 45, pp. 151–158, Feb. 1999.

[17] R. Varalakshmi, V. R. Uthariaraj, "Huffman based conditional access system for key distribution in digital TV multicast," *Multimedia Tools and Applications*, vol. 74, no. 9, pp. 2899–2912, May 2015.

[18] Z. Wan, J. Liu, R. Zhang, R. H. Deng, "A collusion-resistant conditional access system for flexible-pay-per-channel pay-tv broadcasting," *IEEE Transactions on Multimedia*, vol. 15, no. 6, pp. 1353–1364, 2013.

[19] J. Wei, J. Liu, Y. Liu, C. Wei, "A novel entitlement management message distribution for conditional access system," in *Proceedings of 1st International Symposium on Computer Network and Multimedia Technology (CNMT'09)*, pp. 1–4, 2009.

[20] X. Zhao and F. Zhang, "A new type of ID-based encryption system and its application to pay-TV systems," *International Journal of Network Security*, vol. 13, no. 3, pp. 161–166, 2011.

Shih-Ming Chen received the B.S. degree in Information Management from Chaoyang University of Technology (CYUT), Taichung, Taiwan, Republic of China, in 1999; the M.S. in Information Management from Chaoyang University of Technology (CYUT), Taichung, Taiwan, Republic of China, in 2003. He is currently pursuing his PhD degree in Computer Science and Information Engineering from Asia University. His current research interests include information security and Science & Technology of Chinese studies.

Min-Shiang Hwang received the B.S. in Electronic Engineering from National Taipei Institute of Technology, Taipei, Taiwan, Republic of China, in 1980; the M.S. in Industrial Engineering from National Tsing Hua University, Taiwan, in 1988; and the Ph.D. in Computer and Information Science from National Chiao Tung University, Taiwan, in 1995. He also studied Applied Mathematics at National Cheng Kung University, Taiwan, from 1984-1986. Dr. Hwang passed the National Higher Examination in field "Electronic Engineer" in 1988. He also passed the National Telecommunication Special Examination in field "Information Engineering", qualified as advanced technician the first class in 1990. From 1988 to 1991, he was the leader of the Computer Center at Telecommunication Laboratories (TL), Ministry of Transportation and Communications, ROC. He was also a project leader for research in computer security at TL in July 1990. He obtained the 1997, 1998, and 1999 Distinguished Research Awards of the National Science Council of the Republic of China. He is a member of IEEE, ACM, and Chinese Information Security Association. His current research interests include database and data security, cryptography, image compression, and mobile communications.

A Public-Key Approach of Selective Encryption for Images

Feng Jiang, Paul Salama, and Brian King
(Corresponding author: Brian King)

Department of Electrical and Computer Engineering, Indiana Univeristy-Purdue University Indianapolis
420 University Blvd, Indianapolis, IN 46202, USA
(Email: briaking@gmail.com)

(Received Oct. 3, 2015; revised and accepted Dec. 7, 2015 & Feb. 28, 2016)

Abstract

Data security protection is essential for most multimedia data transmissions today. Classical multimedia content protection is dominated by symmetric encryption methods. This paper explores the possibility of public-key media content encryption. The key problem to the asymmetric selective encryption is formalized and defined as the "bounded plaintext problem". Possible solutions to this problem are proposed. A public-key multimedia encryption model is developed and the implementation results are provided.

Keywords: Bounded plaintext problem, EZW, public-key encryption, selective encryption, SPIHT

1 Introduction

Today, society has an insatiable desire for multimedia (such as video or images). The computational and bandwidth demand of multimedia can be significant. Multimedia data usually requires enormous storage and real time computation capabilities. Consequently, efficient multimedia compression techniques are necessary. Further, multimedia content is frequently transmitted over an insecure network [8, 37]. Many applications like military image databases, video conference, medical imaging system, etc. require efficient and secure digital image transmissions [4, 6, 12, 13]. Encryption is the essential technology used to provide confidentiality of the multimedia content. One typical multimedia encryption technique is to merge the compression and encryption methods together, a process called *selective encryption* [19, 25, 32, 35].

The classical multimedia encryption systems usually utilizes symmetric encryption schemes or hybrid encryption [5, 7] schemes. The latter encrypts the symmetric encryption key using public-key encryption schemes [33] and then encrypts the multimedia data using symmetric key encryption. If a public-key encryption scheme is used to protect the multimedia data sequence, then the encryption process needs to be implemented repeatedly, because of the capacity of a single public-key encryption. This adds a significant computational burden on the sender, more so for the receiver.

Using the selective encryption, a desired security can be achieved by encrypting part of the multimedia data. Based on Uhl's [21, 22] analysis of JPEG2000, at least 20% of the image data need to be encrypted to achieve a reliable security. Brahimi [2] showed in his work that 11% of a medical image encrypted leads to a suitable PSNR (indicating low quality). Lian [18] presented a result that 15.3% of the image data encrypted generates a poor decompressed image, which is secure for common applications. However, these plaintext sizes significantly exceed the common plaintext sizes of the public-key encryption (as listed in Table 1). It is accepted by most scholars [21, 22, 39] that the public-key encryption schemes should not be used for multimedia content protection.

In networked applications of multimedia delivery, the symmetric key is usually transmitted from the receiver to the sender using some public-key cryptographic scheme. Thus, at least one public-key cryptographic algorithm must be applied, even when the multimedia content is protected using symmetric key encryption. It would be much more efficient if one can deliver the secured multimedia content using only one public-key encryption. Motivated by this, we explored the possibility of public-key multimedia encryption in this paper. Here we formally define a problem, the "bounded plaintext problem" (see Definition 1), which will be the essential problem of using public-key methods for multimedia encryption. We discuss a solution, as well as results from our implementation experiments.

The rest of the paper is organized as follows. In Section 2, the bounded plaintext size problem is defined based on public-key cryptosystems and the selective encryption for multimedia data. Section 3 discusses the solution to the bounded plaintext problem. Experimental results are provided in Section 4. In Section 5 we provide a conclusion.

2 The Bounded Plaintext Problem

Common public-key cryptosystems include: RSA, ElGamal, Elliptic Curve El Gamal, etc. In a public-key cryptosystem there are two keys, a public key and a secret key. The two keys are mathematically related, but it is "computationally infeasible" to determine the secret key from the public-key [33]. The "feasibility" is related to a computationally "hard" problem. Many of the public-key cryptosystems are related to one of three hard problems: the integer factoring problem, the discrete-log problem and the elliptic curve discrete log problem.

Typically the security of a secure cryptosystem is based on a time period. That is, if there is no known attack on a cryptosystem, then the only remaining attack is some type of key search (such as a factoring algorithms, Pohlig-Hellman, etc.). Such algorithms tend to run in exponential time and are useful when they are able to run for sufficient amount of time or when the key is small. There has been several studies concerning public-key cryptosystems and the appropriate keysize for a given time period [9, 11, 14, 16].

A parameter that determines the security of a secure public-key cryptosystem is typically a fixed year, such that the time difference between the end of the fixed year and the current date provides a duration for a key search algorithm to determine the key. Thus, once one fixes a year, then a lower bound on the key size is provided.

We refer to a secure plaintext size as the "bound" on the plaintext size for keys secured until a given year. The secure plaintext size for these three problems, for the year 2015, is given in Table 1. Here, the problems have been analyzed with several analysis approaches such as Lenstra [16], etc.

Table 1: Plaintext length (bits) of public-key encryption secured for the year of 2015

Analysis Approach	Factoring	Discrete Logarithm	Elliptic Curve Logarithm
Lenstra [16]	1248	1613	154
Lenstra [14] Updated	1350	1245	156
ECRYPT II [11]	1248	1248	160
NIST [1]	2048	2048	224
ANSSI [9]	2048	2048	200

The "hardness of a problem" is often measured against the problem (input) size. For example RSA is related to the integer factoring problem. Currently, factoring a 1248 bit composite integer is considered infeasible. Consequently a secure plaintext length for RSA is approximately 1248 bits. In general, for a given public-key cryptosystem, if one fixes the parameter(s) to a security level for the underlying hard problem, then one has bounded the plaintext size to the given level[1].

[1]By security level, we are referring to a period of time or year, for which the secret key is secure against any known attacks.

For example, if once we adopt 1248 bits RSA in our multimedia data encryption, then we can support plaintext size up to 1248 bits. Throughout we will use Δ to represent the security level parameter.

Selective encryption ($\mathcal{SE}$) [19, 25, 32, 35] has been proposed and applied to multimedia encryption by scholars in recent years. The concept of selective encryption is to selectively encrypt part of the multimedia data. The two basic parts of selective encryption are the selection process S_{proc} and the encryption process E_{proc}. The E_{proc} is considered to be a typical cryptographic algorithm encrypting the plaintext M, where M is a subset of the multimedia sequence Seq. For example, in [24], a certain bitplane of the image sequence is selected to encrypt, the subset M, in this case, is the selected bitplane from all bitplanes.

It was shown by Hellman [10] that compressing data before encrypting it, increases security in the sense that more ciphertext will be needed to determine the encryption key than if the data had been encrypted directly. To improve the encryption efficiency, the selective encryption is usually performed in some transformed signal domain and integrated within the compression process. For example, [25, 40] implemented the encryption in the frequency domain, [19, 35, 38] integrated the selective encryption with compression techniques. In our previous work[28], only the beginning part of the compressed image sequence is encrypted to increase the encryption efficiency.

An efficient selective encryption is a selective encryption scheme:

$$\mathcal{SE} = (S_{proc}, E_{proc}, \Delta, Comp) \quad (1)$$

where selection process S_{proc} is accomplished in some domain D_m (multimedia compressed sequence), after using compression technique $Comp(\cdot)$, such that the most informative part of the original data is selected and encrypted generating a required security level Δ. Here, E_{proc} is the encryption process, $\mathcal{M}_{se}$ is the plaintext space and $\mathcal{C}_{se}$ is the cyphertext space.

As noted earlier, in a selective encryption scheme the encryption scheme used will be selected to satisfy the security parameter Δ.

However, to evaluate the security of the selectively encrypted data sequence (ciphertext), the whole sequence needs to be evaluated. We recognize that part of the sequence remains in the clear, so one cannot claim the data sequence is secure at a level of Δ. One will have to explicitly design the selection process S_{proc} and evaluate the security of the resulting data sequence accordingly.

When we use the term a *secure selective encryption scheme*, we mean that the entire multimedia data sequence, which includes the selectively encrypted data as well as the data that has not been encrypted, is secure against the known selective encryption attacks.

For example, $Comp$ = JPEG 2000, S_{proc}=first 20 % of the multimedia compressed sequence and E_{proc} =128 bit AES. The key size 128 for symmetric key is secure according to [9] until year 2090, so security parameter $\Delta = \textit{year 2090}$.

Definition 1. ***Bounded Plaintext Problem:*** (E, Δ) *Given a public-key encryption algorithm E and a desired security level Δ, the plaintext size is bounded in bit length. For these fixed parameters, does there exist a secure selective encryption scheme $\mathcal{SE} = (S_{proc}, E_{proc}, \Delta, Comp)$ in which the encryption process E_{proc} is the public-key encryption algorithm E that is applied exactly once.*

For example, if one is given $E_{proc} = RSA$ and security parameter $\Delta = \textit{year 2015}$, then the plaintext size is limited to 1248 bits according to [16]. Thus one can only encrypt 1248 bits of the image. We have seen, via literature, that the compression scheme JPEG 2000 requires a substantial percentage of the image to be encrypted [21, 22]. Thus for (E, Δ)=(RSA,2015), if there exists a secure selective encryption scheme then the compression scheme cannot be JPEG 2000.

The essential task to solve the bounded plaintext problem is to determine the compression scheme $Comp(\cdot)$, which generates very high information rate [23] in a small locality, where the remaining data is provides significantly less information and then determine S_{proc}.

3 Solving The Bounded Plaintext Problem

3.1 Rate-scalable Wavelet Compression

A rate scalable compression [28, 31] technique allows compressing the multimedia data once and decompressing it at multiple data rates or quality. The user can stop the decompression process at any point of the compressed bitstream. A higher bit rate or bandwidth leads to a better decompression quality. There are several rate-scalable compression techniques, such as Embedded Zero tree Wavelet (EZW), Set Partitioning In Hierarchical Trees (SPIHT) and JPEG2000 [17]. EZW and SPIHT compressions are typical implementations of the initial rate scalable theory. Given a rate scalable compressed multimedia sequence, the more bitstream the user can decompress, the higher quality the user can achieve. The rate scalable compression process guarantees the user always get the "best" representation of the multimedia data with limited resources or computing capability. Typical rate scalable compression techniques are developed based on wavelet transform of the original data. The compression is performed to reduce the data dependency within the wavelet transformed frequency domain. As the inherent multi-resolution [27, 30] characteristic of the wavelet transform, the compression process is implemented from the most significant element to the least significant element successively. It is reasonable to expect that the information rate is much higher at the beginning part of the compressed sequences. We first performed the information rate test by using the EZW compression and then implemented both EZW and SPIHT compressions in our pilot encryption/decryption system.

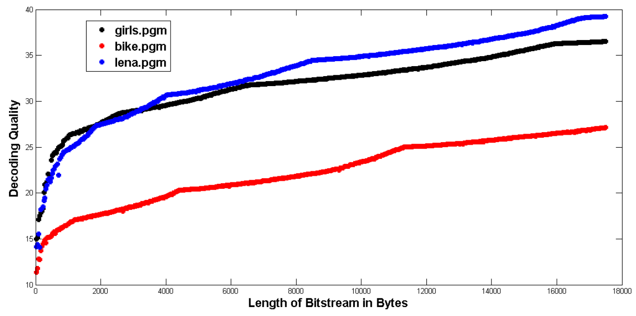


Figure 1: Decompression quality change through the whole bitstream

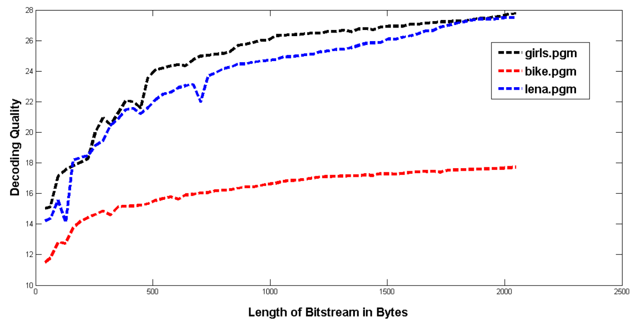


Figure 2: Decompression quality change within the first 2048 bytes

3.2 Testing Rate-scalable Wavelet Compression as a Solution to Bounded Plaintext Problem

3.2.1 Information Intensity Test

It is well know that the decompression quality increases by increasing decompression bitstream length. To test the information rate of the compressed bitstream, we successively increased the length of the selection of the bitstream and feed that into the decoder. The decoded image quality is calculated and plotted. As shown in Figure 1, Figure 2, three different images are tested, the decompression quality is gradually increased by increasing the decoding bitstream length. However, the decompression quality is not increased in a constant speed. Figure 2 is a concentrated version of Figure. 1. The decompression quality is increased greatly by only decoding the first 2048 bytes of the whole bitstream and the decompression quality increases at a much higher speed when the decoding bitstream length is small. Therefore, it is possible to find a small part of the entire bitstream representing the most significant information of the entire image. Our hypothesis was proven to be practical.

The experimental result also confirms the characteristics of wavelet compression in terms of applying such compression technique in this setting. Usually, the low frequency coefficients of the image are compressed first and the marked as the most important elements. The decoder can retrieve the background color of the image first and retrieve the detailed image features successively in the decoding process. The information rate at the beginning part of the compressed bitstream is much higher than the other parts.

3.2.2 Correlation Test

However, it is still too early to argue that hiding the initial part of the bitstream will secure all the useful information of the image, if it is possible to recover the beginning part by any other correlated bitstream. Our next experiment tested the correlation between a selected bit sequence at the initial part and a later part of the compressed sequence.

The first 224 bits (excluding the header part) of the compressed sequence is selected as a target sequence according to the public-key plaintext size in the Table. 1. To fully test the correlation between the target sequence and all the other bit sequences within the bitstream, we sampled through the entire bitstream uniformly and generated 250 sequences with same bit length as the selected targeted sequence. The correlation test was performed for three 512×512 grayscale images, "lena.pgm", "girls.pgm" and "bike.pgm". The goal of the correlation test is to check the relationship between two bit sequences in bit length n, denoted as bitstream X and bitstream Y.

In our experiment, we noticed that the general statistical correlation calculation

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (2)$$

was not a good fit here because the samples x_i and y_i can only take on value "0" or "1" in a bitstream. Thus different bitstream pairs with different patterns but same hamming weights will generate same correlation values. Consequently, we adopted the correlation coefficient C_{XY} test designed for bit sequences as described by Menezes et. al. in [20]. For $X = x_1, \ldots x_n$ and $Y = y_1, \ldots, y_n$, C_{XY} is defined as

$$C_{XY} = \frac{2(A(X,Y) - \frac{n}{2})}{\sqrt{n}}, \quad (3)$$

where

$$A(X,Y) = \sum_{i=1}^{n}(x_i \oplus y_i). \quad (4)$$

Here $A(X,Y)$ represents the Hamming distance between X and Y, $\oplus$ denotes the XOR operator and $\sum$ denotes the summing of bits which differ between X and Y.

The correlation test result C_{XY} approximately follows a Normal $N(0,1)$ distribution when $n \geq 10$ [20]. A two-sided test is applied. The hypothesis, "the two sequence are uncorrelated" serves as the null hypothesis H_0, and the alternative hypothesis, "the two sequences are correlated" is denoted by H_1.

The correlation between the target sequence X and the other 250 sampled sequences generated from the data in clear were tested. Among the 750 correlation values for the three testing images, the largest correlation value is 3.07 in absolute value, 747 out of the 750 absolute correlation values are lower than 2.5758. According to the standard normal distribution $N(0,1)$ [36], with the significance level $\alpha = 0.005$, we will rarely reject the null hypothesis, i.e. rejecting that the two tested sequences are uncorrelated. Thus, one can be very confident that the high information sequence is not correlated with the sequence generated from the data in the clear.

Table 2: Absolute correlation value between the selected bitstream and other bitstreams

image	largest absolute correlation	average absolute correlation
lena	2.539	0.8128
girls	3.0535	0.8054
bike	3.0735	0.77

Table 3: Plaintext length and decompression image quality of EZW sequences

image	size	plaintext length (bytes)	PSNR
lena	512×512	16	7.79
lena	512×512	16	7.77
lena	512×512	16	8.24

3.2.3 Decompression Quality Test

We then tested the decompressed image quality. We applied the decompression process to a sequence consisting of the image bitstream but where a small part of the initial bitstream is scrambled. As shown in Table 3, if a small part of the sequence is corrupted then this could cause severe intelligibility loss.

From the above experiments, we discovered that the rate scalable compression technique, such as EZW, generates a bitstream with decreasing information rate. In fact, the SPIHT compression technique, which is not discussed in this section, has proven to have the same characteristic.

It is included in the implementation results. The information rate is highest at the initial part of the bitstream. A small part of the bitstream lost or ruined in this part destroys the decompression process and the lost part cannot be recovered by any of the remaining bits of the bitstream.

3.3 Using Public-key Cryptosystems in a Selective Encryption Scheme

The encryption and decryption process of our public-key selective encryption model can be represented by $(E, D, \mathcal{K}_{sk}, \mathcal{K}_{pk}, \mathcal{M}, \mathcal{C})$ with the secret key space $\mathcal{K}_{sk}$ and public key space $\mathcal{K}_{pk}$. E is the encryption algorithm of selective encryption $\mathcal{SE}$ we defined in Equation (1). D is the compression output sequence of the compression technique *Comp* defined in Equation (1). The plaintext $\mathcal{M}$ is one plaintext of the plaintext space $\mathcal{M}_{se}$ defined in Equation (1) and Δ is the required security level. Let n denote the plaintext bound in bits and h denote the number of bits of the parameters within the header that need to be encrypted. The procedures of encryption and decryption are described in Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 Encryption process of the public-key selective encryption

```
Public key encryption: $(E, D, \mathcal{K}_{sk}, \mathcal{K}_{pk}, \mathcal{M}, \mathcal{C})$
user: public key pk
Security parameter: $\Delta$ which implies $n$ bit plaintext bound
input: Image $I$
1: $D = Comp(I)$
2: $H = header$ of $I$ and $h$ bits of $H$ need to be secured
3: For bit $b \in D \cup H$,
     $S_{proc}(b) = \begin{cases} 1 & b \in H \text{ and } b \text{ needs to be secured} \\ 1 & b \in D \text{ and b is among first } n-h \text{ bits} \\ 0 & \text{otherwise} \end{cases}$
4: Let $\mathbf{b} = (b_{1_1}, \ldots, b_{i_n})$ where $S_{proc}(b_{i_j}) = 1$.
5: Map $\mathbf{b}$ to $m_{\mathbf{b}} \in \mathcal{M}$
6: Let $\mathbf{r}$ denote the set $(\nu_1, \ldots \nu_t)$, $\nu_i \in D \cup H$ which $S_{proc}(\nu_i) = 0$.
7: Let $\mathbf{o} = (E_{\mathsf{pk}}(m_{\mathbf{b}})||\mathbf{r})$
8: Return the selective encryption ciphertext $\mathbf{o}$
```

The public-key encryption algorithm we applied our work to is elliptic curve cryptography, which is well recognized as a bandwidth friendly type of public-key encryption. To form a general solution of the bounded plaintext problem, we select a public-key algorithm (elliptic curve encryption) with relatively small plaintext size.

Elliptic curve cryptography is defined over some finite field $\mathbb{F}$. For $a_i \in \mathbb{F}$ for $i = 1, ...5$, the elliptic curve E is a set of all points in $\mathbb{F} \times \mathbb{F}$, which satisfy an equation of the form

$$y^2 + a_1xy + a_2y = x^3 + a_3x^2 + a_4x + a_5, \qquad (5)$$

as well as the point of infinity $\mathcal{O}$.

There is a natural addition defined over E and so that E forms a finite additive abelian group, and the curve is selected so that a large prime q divides the group order of E.

Algorithm 2 Decryption process of the public-key selective encryption

```
Public key encryption: $(E, D, \mathcal{K}_s, \mathcal{K}_p, \mathcal{M}, \mathcal{C})$
user: secret key sk
Security parameter: $\Delta$ which implies $n$ bit plaintext bound
input: ciphertext $\mathbf{o}$
1: Let $\mathbf{c}$= first $n$ bits of $\mathbf{o}$ and let $\mathbf{r}$ = remaining bits of $\mathbf{o}$
2: Let $m = Dec_{\mathsf{sk}}(\mathbf{c})$
3: Map $m \in \mathcal{M}$ to a bit sequence $\mathbf{b}$
4: Let $\mathbf{h}$ be the first $h$ bits of $\mathbf{b}$ and let $\mathbf{w}$ denote the remaining bits
5: Insert $\mathbf{h}$ and $\mathbf{w}$ into $\mathbf{r}$ in the appropriate places, the result is $\mathbf{v}$. This step is defined by $S_{proc}$
6: Let $T = Decompress(\mathbf{v})$
7: Return image $T$
```

The scalar multiple of a fixed point P of E is the necessary cryptographic applications. That is, k is a positive integer $0 < k < q$ and the scalar multiple kP (which is $P + P + \cdots P$) is the essential calculation.

To achieve elliptic curve encryption one can use *Elliptic Curve El Gamal* [34]. If k is the user's secret key then kP is the public key. In order to encrypt the message W (which is an elliptic curve point), the sender gets the user public key kP, they select r randomly, where $0 < r < q$. They then compute

$$C_1 = rP \text{ and } C_2 = W + r(kP).$$

They then send (C_1, C_2) to the user. The user can decrypt (C_1, C_2) and can recover W by calculating

$$W = C_2 - kC_1.$$

According to Table 1, 224 bits plaintext size should be secure for elliptic curve encryption today. So q is approximately 224 bits in size. Thus elements in $\mathbb{F}$ are bounded by 224 bits. Though the message $M \in E \subset \mathbb{F} \times \mathbb{F}$, is a point , with approximately 448 bits. Because M satisfies Equation (5), the y-coordinate of the point has very little entropy (approximately one bit). Thus for this setting, the bounded plaintext problem is such that the plaintext is bounded by 224 bits. Concerning the conversion of the image sequence b to a point W. It is not trivial to map a bit-sequence to a point in an elliptic curve. Work in [15] and [34] show how such a mapping can be performed. The details of this mapping is outside the scope of this paper.

4 Analysis of the Implementation Results

Our analysis shows that it is difficult to retrieve any information from the compressed sequences with only a small part of the multimedia data sequence encrypted. Thus the bounded plaintext problem can be solved by utilizing a wavelet rate-scalable compression technique, such as EZW and SPIHT. The public-key multimedia encryption can be achieved by selectively encrypting a small important part of the multimedia data sequence. Here, the encryption process E_{proc} can be elliptic curve public-key encryption. The plaintext, in our work is bounded in 224 bits in order to utilize a elliptic curve public-key encryption secured for 2015.

To select the "most important part" from the multimedia sequence generated by compression $Comp(\cdot)$, the image sequence is first analyzed by per element.

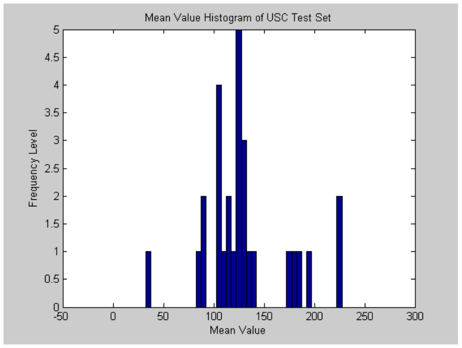


Figure 3: Image mean value distribution map of the USC test set

Usually there are two parts of a compression sequence, the general encoding settings (the header part) and the data part. For the header part, coding parameters such as compression rate, input output image dimensions, quality requirement and package indicators are not considered as necessary elements to be kept private, so no need to encrypt.

However, some important parameters related with image content or unique image identification information are necessary to be protected or tested to see if they should be protected.

For example, the initial threshold value and the mean value of the image in the EZW compression should be encrypted. The DC component of the wavelet transform in SPIHT compression should also be encrypted. The encoding threshold or DC component is considered as a necessary element because it is determined by both the original image features and the wavelet transform. Leaving the image mean value in the clear may result in leaking important information such as the background color or image shooting time. As shown in Figure. 3, it is natural to expect most images have the mean value near the middle point of the entire gray level, which is 128. However, an image with very dark or white background laying on the side parts of the map will leak information easily.

Both EZW and SPIHT compression techniques were implemented in our experiments. To satisfy the bounded plaintext requirement for the elliptic curve public-key encryption, 224 bits are selected from each rate scalable compression sequence. As listed in Table 4, for the EZW compression, the initial threshold, image mean value and

Table 4: Plaintext selection of EZW and SPIHT sequences

	Plaintext reconstruction	Bit location
EZW	image mean within the header	56 - 63
EZW	initial threshold value	64 - 71
EZW	package data	72 - 279
SPIHT	DC component within the header	105 - 112
SPIHT	package data	113 - 328

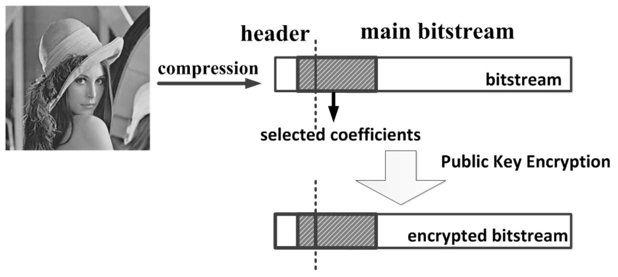


Figure 4: Proposed asymmetric encryption model-encryption process

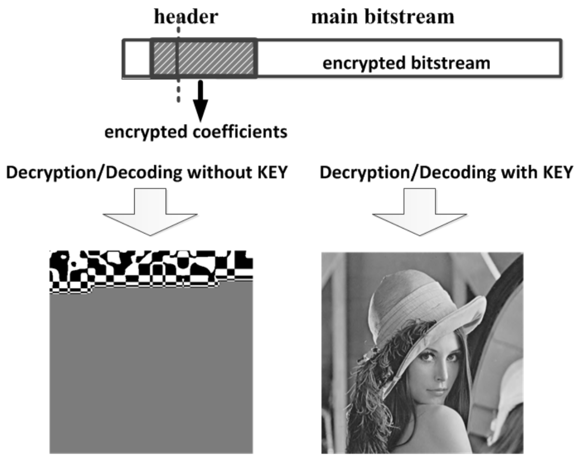


Figure 5: Proposed asymmetric encryption model-decryption process

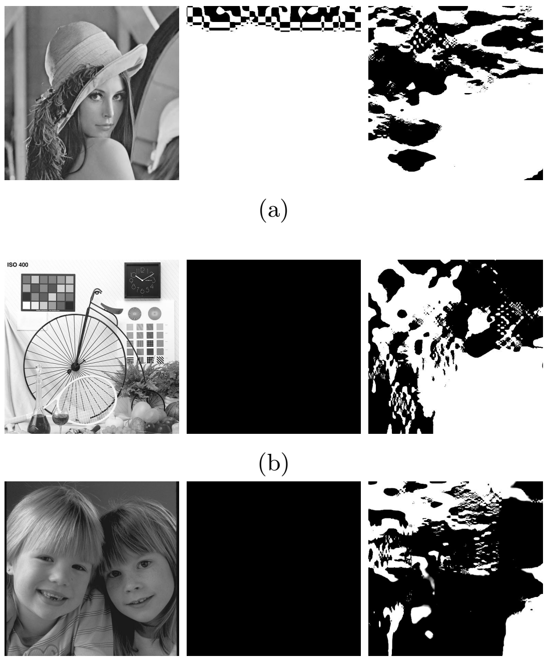
Table 5: Data replacement attack of encrypted bit streams

PSNR	Bike	Girls	Lena
Inserting zeros	10.6713	9.6280	11.8726
Inserting ones	7.8563	14.8034	13.4597
Shifting over	10.3656	14.8034	14.3145
Inserting another bit-stream	9.6015	14.8034	12.0017

the followed sequence data are selected to form 224 bits plaintext. 55 bits header information are skipped. For the SPIHT compression, all the header information except the DC component are skipped. The plaintext starts at the 105th bit. The reference EZW compression codec used in our experiment was developed by video and image processing laboratory at Purdue University [3, 26, 28, 29].

Table 6: Encryption data ratio compared with other existing methods

Method	Image	Encryption data ratio	Plaintext length (byte)	PSNR of decompressed image	Number of ECC encryption
Uhl[22, 21]	angiogram 512 × 512	20%	6554	9.90	234
Uhl[22, 21]	lena 512 × 512	20%	6554	9.77	234
Lian[18]	lena 128 × 128	13.4%	274	N/A	10
Lian[18]	village 512 × 512	16.8%	5505	N/A	197
Lian[18]	boat 256 × 256	10.8%	885	N/A	32
Brahimi[2]	radiological 256 × 256	11.6%	954	7.47	34
Salama[28]	bike 512 × 512	0.1%	32	9.48[2]	1.14
Salama[28]	barbara 512 × 512	0.1%	32	14.08[3]	1.14
Proposed Method	lena 512 × 512	0.09%	28	5.06	1
Proposed Method	girls 512 × 512	0.09%	28	7.34	1
Proposed Method	bike 512 × 512	0.09%	28	2.14	1


(a)

(b)

(c)

Figure 6: Decrypted images of EZW (middle) and SPIHT (right) sequences with and without a correct secret key (a) “lena” (b) “bike” (c) “girl”

The reference SPIHT compression codec used is developed by René Puchinger.

The compression bit rate is 1 bit per pixel for both EZW and SPIHT compression.

As shown in Figure 4 and Figure 5, the encryption process of our asymmetric encryption model reads and compresses one original image. The sequence analysis is performed in the compression process. Essential elements (h *bits*) within the header and the first $224-h$ bits within the data sequence are marked and then encrypted by the elliptic curve public-key encryption.

The decryption process reads the encrypted sequence and performs the sequence analysis to decide the plaintext location. The public-key decryption algorithm is then performed to recover the plaintext and merge it into the compression sequence. The decompression process generates a desired image or a severely destroyed image if the public-key decryption is not performed by a correct secret key. As listed in Figure 6, the decryption output images without a correct secret key represents no useful information at all.

To confirm the encrypted sequences provide security with a small plaintext length, the data replacement attack is tested also.

We assume the eavesdropper obtained an encrypted sequence with a small cyphertext, they can apply the data replacement [24] attacks to it by simply inserting constant bits or other sequence data. We performed several typical data replacement such as inserting zero, inserting one, shifting over and inserting another bitstream to the encrypted sequence. Table 5 illustrates the decompression image quality of these data replacement. The plaintext length is 224 bits, image size is 512×512, “bike”, “girls” and “lena” testing images were tested. The poor decompression quality proved again that the encrypted part of the bitstream represented most important information of the image. Simple data replacements does not reveal any information.

To compare our proposed method with all the existing methods, the encryption parameters are listed in Table 6. Different images are selectively encrypted by different methods to be transmitted securely. The extremely poor reconstruction quality indicated that a satisfied security level can be achieved by encrypting by different data ratios. (Notice that, poor reconstruction quality can be shown by low PSNR or visually illustrated as well. The “N/A” in Table 6 means the reconstruction quality was visually illustrated but no PSNR values were measured.) The plaintext sizes are calculated by assuming all the images are compressed to one bit per pixel. Notice that the smaller size images require much lower plaintext size using same method. For the images in same size, our previous work [28] and this proposed method achieved good security by using much lower plaintext sizes. To apply common public-key encryption algorithms, such as elliptic curves encryption, our proposed method is the only one that enables public-key media content encryption without encrypting the original data repeatedly.

Common traditional media content encryption [2, 18, 21, 22, 28], simply selecting part of (for example, 20%) the original media content, are not designed to cooperate with asymmetric encryption. The plaintext size dramatically increases as the original image size increases. Targeting at finding only the essential part of the original image content,

Here we have formalized the key problem to asymmetric media content encryption and have solved it for the first time, which makes asymmetric media encryption practical.

5 Conclusion

As far as we know, this paper is the first paper that advocates using selective multimedia encryption with a public-key encryption scheme.

All other work in the selected encryption area have advocated to use symmetric encryption or some hybrid approach (use of public-key and symmetric key cryptosystems).

The key problem to asymmetric media content encryption has been formally defined and analyzed. Solutions to this problem have been proposed. The encryption security was tested by multiple methods such as image quality test, sequence correlation test and cryptographic security analysis. The public-key selective multimedia encryption model was proposed based on elliptic curve public-key encryption and two rate scalable compression techniques. Experimental results confirmed a successful implementation of a secured public-key selective multimedia encryption.

References

[1] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, *Recommendation for Key Management, Part 1: General (revised)*, NIST Special Publication, Citeseer, 2006.

[2] Z. Brahimi, H. Bessalah, A. Tarabet, and M. K. Kholladi, "A new selective encryption technique of JPEG2000 codestream for medical images transmission," in *IEEE 5th International Multi-Conference on Systems, Signals and Devices (IEEE SSD'08)*, pp. 1–4, 2008.

[3] Edward J Delp, Paul Salama, Eduardo Asbun, Martha Saenz, and Ke Shen, "Rate scalable image and video compression techniques," in *IEEE 42nd Midwest Symposium on Circuits and Systems*, vol. 2, pp. 635–638, 1999.

[4] N. F. El Fishawy and O. M. Abu Zaid, "Quality of encryption measurement of bitmap images with RC6, MRC6, and rijndael block cipher algorithms," *International Journal of Network Security*, vol. 5, no. 3, pp. 241–251, 2007.

[5] D. S. Abd Elminaam, H. M. Abdual-Kader, and M. M. Hadhoud, "Evaluating the performance of symmetric encryption algorithms," *International Journal of Network Security*, vol. 10. no. 3, pp. 216–222, 2010.

[6] J. B. Feng, I. C. Lin, C. S. Tsai, and Y. P. Chu, "Reversible watermarking: current status and key issues," *International Journal of Network Security*, vol. 2, no. 3, pp. 161–170, 2006.

[7] E. Fujisaki and T. Okamoto, "Secure integration of asymmetric and symmetric encryption schemes," in *Advances in Cryptology (CRYPTO'99)*, pp. 537–554, Springer, 1999.

[8] B. Furht, D. Socek, and A. M. Eskicioglu, *Fundamentals of Multimedia Encryption Techniques*, Multimedia Security Handbook, 2004.

[9] D. Giry and P. Bulens, "Keylength–cryptographic key length recommendation," 2009. (`http://www.keylength.com`)

[10] M. E. Hellman, "An extension of the shannon theory approach to cryptography," *IEEE Transactions on Information Theory*, vol. 23, no. 3, pp. 289–294, 1977.

[11] ECRYPT II and D. SYM, "Ecrypt II," 2011.

[12] I. A. Ismail, M. Amin, and H. Diab, "A digital image encryption algorithm based a composition of two chaotic logistic maps," *International Journal of Network Security*, vol. 11, no. 1, pp. 1–10, 2010.

[13] X. Kang, W. Zeng, and J. Huang, "A multi-band wavelet watermarking scheme," *International Journal of Network Security*, vol. 6, no. 2, pp. 121–126, 2008.

[14] A. K. Lenstra, *Key Lengths*, The Handbook of Information Security, Springer, 2004. (`https://infoscience.epfl.ch/record/164539/files/NPDF-32.pdf`)

[15] B. King, "Mapping an arbitrary message to an elliptic curve when defined over gf (2^n)," *International Journal of Network Security*, vol. 8, no. 2, pp. 169–176, 2009.

[16] A. K. Lenstra and E. R. Verheul, "Selecting cryptographic key sizes," in *Public Key Cryptography*, LNCS 1751, pp. 255–293, Springer, 2001.

[17] Z. N. Li, M. S. Drew, and J. Liu, "Image compression standards," in *Fundamentals of Multimedia*, pp. 281–315, Springer, 2014.

[18] S. Lian, J. Sun, D. Zhang, and Z. Wang, "A selective image encryption scheme based on JPEG2000 codec," in *Advances in Multimedia Information Processing (PCM'04)*, pp. 65–72, Springer, 2005.

[19] X. Liu and A. M. Eskicioglu, "Selective encryption of multimedia content in distribution networks: Challenges and new directions," in *Communications, Internet & Information Technology*, pp. 527–533, 2003.

[20] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC press, 1996.

[2]Numbers are read from plotes in the paper [28].

[3]Numbers are read from plotes in the paper [28].

[21] R. Norcen, M. Podesser, A. Pommer, H. P. Schmidt, and A. Uhl, "Confidential storage and transmission of medical image data," *Computers in Biology and Medicine*, vol. 33, no. 3, pp. 277–292, 2003.

[22] R. Norcen and A. Uhl, "Selective encryption of the JPEG2000 bitstream," in *Communications and Multimedia Security*, LNCS 2828, pp. 194–204, Springer, 2003.

[23] J. Pieprzyk, T. Hardjono, and J. Seberry, *Fundamentals of computer security*, Springer Science & Business Media, 2013.

[24] M. Podesser, H. P. Schmidt, and A. Uhl, "Selective bitplane encryption for secure transmission of image data in mobile environments," in *Proceedings of the 5th IEEE Nordic Signal Processing Symposium (NORSIG'02)*, pp. 4–6, 2002.

[25] A. Pommer and A. Uhl, "Selective encryption of wavelet-packet encoded image data: Efficiency and security," *Multimedia Systems*, vol. 9, no. 3, pp. 279–287, 2003.

[26] M. Saenz, P. Salama, K. Shen, and E. J. Delp, "Evaluation of color-embedded wavelet image compression techniques," in *Electronic Imaging*, pp. 282–293, 1998.

[27] A. Said, W. Pearlman, et al, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243–250, 1996.

[28] P. Salama and B. King, "Efficient secure image transmission: compression integrated with encryption," in *Electronic Imaging*, pp. 47–58, 2005.

[29] P. Salama, N. Shroff, and E. J. Delp, "Error concealment in embedded zerotree wavelet codecs," in *Proceedings of the International Workshop on Very Low Bit Rate Video Coding*, pp. 8–9, 1998.

[30] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *SIEEE Transactions on Ignal Processing*, vol. 41, no. 12, pp. 3445–3462, 1993.

[31] K. Shen and E. J. Delp, "Wavelet based rate scalable video compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 1, pp. 109–122, 1999.

[32] T. Shi, B. King, and P. Salama, "Selective encryption for H.264/AVC video coding," in *Proceedings of SPIE*, vol. 6072, pp. 461–469, 2006.

[33] D. R. Stinson, *Cryptography: Theory and Practice*, CRC press, 2005.

[34] W. Trappe and L. C. Washington, *Introduction to Cryptography with Coding Theory*, Pearson Education India, 2006.

[35] M. V. Droogenbroeck and R. Benedett, "Techniques for a selective encryption of uncompressed and compressed images," in *Advanced Concepts for Intelligent Vision Systems (ACIVS'02)*, pp. 90–97, 2002.

[36] D. Wackerly, W. Mendenhall, and R. Scheaffer, *Mathematical statistics with applications*, Cengage Learning, 2007.

[37] C. P. Wu and C. C. Jay Kuo, "Efficient multimedia encryption via entropy codec design," in *Photonics West 2001-Electronic Imaging*, pp. 128–138, 2001.

[38] C. P. Wu and C. C. J. Kuo, "Design of integrated multimedia compression and encryption systems," *IEEE Transactions on Multimedia*, vol. 7, no. 5, pp. 828–839, 2005.

[39] T. Xiang, J. Qu, C. Yu, and X. Fu, "Degradative encryption: An efficient way to protect spiht compressed images," *Optics Communications*, vol. 285, no. 24, pp. 4891–4900, 2012.

[40] W. Zeng and S. Lei, "Efficient frequency domain selective scrambling of digital video," *MIEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 118–129, 2003.

Feng Jiang is a PhD student at Purdue West Lafayette. Her research interests include image processing, multimedia security, visual cryptography, and Watermarking.

Paul Salama received his Ph.D. from Purdue University in 1999. Research activities include Signal/image/video processing, biomed. image analysis, image/video compression, security, and restoration/reconstruction, digital communications, information theory and source coding, error resilience, and error concealment.

Brian King received a Ph.D. in Mathematics from University of Wisconsin-Milwaukee in 1990 and a Ph.D. in Computer Science from University of Wisconsin-Milwaukee in 2000. His research interests include Information security, computer & network security, wireless security, cryptography, Ubiquitous & Mobile Ad-hoc Network Security, algorithms, and applied mathematics.

An Efficient Identity-based Online/Offline Signature Scheme without Key Escrow

Dan Liu[1], Shun Zhang[1], Hong Zhong[1], Runhua Shi[1] and Yimin Wang[1,2]
(Corresponding author: Runhua Shi)

School of Computer Science and Technology & Anhui University[1]
Hefei 230601, China
Modern Education and information Center & Anhui Agriculture University[2]
Hefei 230036, China
(Email: shirh@ahu.edu.cn)

(Received Oct. 31, 2015; revised and accepted Jan. 30 & Mar. 29, 2016)

Abstract

Recently, several identity-based signature schemes with Bilinear Pairing and Map-To-Point (MTP) functions have been introduced. However, identity-based cryptography (IBC) schemes suffer from the serious secure problem due to Key Escrow. In addition, both Bilinear Pairing and MTP function are time-consuming operations, and thus the cryptographic schemes based on these expensive operations have high computational burden. In this paper, we proposed an efficient identity-based online/offline signature scheme without using Bilinear Pairing and MTP function. Especially, the proposed scheme overcomes the key escrow problem, and achieves some good features. Furthermore, the securities of the proposed scheme were proven in the random oracle model with the hardness of elliptic curve discrete logarithm problem (ECDLP).

Keywords: Elliptic curve discrete logarithm problem, identity-based, key escrow, online/offline signature, traceability

1 Introduction

Cryptography is an important tool that enables the secure transmission of a secret message between a sender and a recipient from any potential eavesdropper. Furthermore, key management, including key generating, updating, transmitting, storing and so on, is a critical issue for most cryptosystems. Generally speaking, key management is the weakest link in the whole cryptosystem, because key leakage will directly lead to the leakage of plaintext content. Naturally, lots of cryptosystems seeks the help of Key Escrow to deal with key management [14]. Key Escrow is an arrangement in which the keys needed to decrypt the encrypted data are held in escrow so that an authorized third party may gain access to those keys under certain circumstances. For example, once the user's key is lost, according to the established rules, the user can directly get the key from the authorized third party. However, Key Escrow also brings some risks, such as key leakage or misuse, though it has low computational costs in the phase of key generation and distribution.

1.1 Related Works

In 1990, the concept of online/offline signature was first put forward by Even, Goldreich and Micali [8]. The key idea is to split the signature generation algorithm into two phases: the offline phase and the online phase. The signing algorithm executes the offline phase to perform most of the computations and stores without knowing the signed message. Once the signed message is available, the signing algorithm runs the online phase very quickly and requires only light computations. Obviously, Online/offline signatures are more useful in some power or storage limited devices, such as smart card, wireless sensor and RFID tags, because the offline phase can be executed either during device manufacturing process or as a background computation whenever the device is connected to power. Accordingly, some signature schemes can be naturally split into offline and online phases. For instance, the partial computation of some signature schemes (e.g. Schnorr, ElGamal, DSS signature schemes) does not depend on the given message, and thus it can be shifted to the offline phase directly. The first general method for transforming any ordinary signature scheme into online/offline signature scheme was proposed by Even, Goldreich and Micali [8]. Nevertheless, their method is inefficient and impractical because it increases the length of each signature by a quadratic factor. In 2001, Shamir and Thuman [21] constructed an improved online/offline signature scheme which was based on trapdoor hash function. It highly enhances the efficiency, particularly in the online phase. However, it increases the computational costs of signing and there exist trapdoor leak problems. In 2007, to overcome trapdoor leak problems, Chen [6] designed an online/offline signature scheme by utilizing the double trapdoor hash function. But, both schemes are not actually efficient or practical to be used, in generic settings.

In traditional public key cryptosystem (PKC), a random public key of one user is associated with the user by a certificate. And PKC is based on public key infrastructure (PKI), which incurs additional cost for setting up the infrastructure, and requires the public key certificate management and distribution. In addition, the generation, delivery, management, and revocation of the public key certificate need to consume huge storage space and high computational costs. To solve these problems of PKC, Shamir [20] introduced the notion of identity-based cryptography (IBC) where the public key of each user can be gained from some public information (e.g., ID number, IP address and user's name) that exclusively identifies the user and also is known to others users. In IBC, there is a trusted third party (TTP), called Private Key Generator (PKG), who computes the private key from the master secret for the users. The PKG first publishes a master public key, and retains the corresponding master private key in secret. IBC simplifies key agreement procedures of certificate-based PKI. Thenceforward, several identity-based signature schemes [7, 16, 22, 23] have been proposed, which are based on the difficulty assumption that the integer factorization problem (IFP) is hard.

The first identity-based online/offline signature scheme was introduced by Xu, Mu and Susilo [27], which combines the advantages of IBC and online/offline signature. The key certificates are eliminated, and most computations needed for the signature generation are computed before the messages are given, so that the online phase generates the signature efficiently after the messages are available. In 2006, the same authors [26] designed another efficient identity-based online/offline signature scheme for authentication in AODV routing protocol. Nevertheless, Ming et al. [28] showed that the scheme in [27] is universally forgeable. And Li et al. [15] proved that the scheme in [26] does not achieve the security. To overcome the drawback, various improved schemes have been proposed. For instance, in 2014, Kar [13] introduced a new identity-based online/offline signature scheme. Unfortunately, the scheme of Kar has high computational costs because of bilinear pairing and map-to-point (MTP) function.

1.2 Motivations and Contributions

In the identity-based signature schemes based on traditional PKI, there is a TTP, called PKG, who computes the private key from the master secret for the users. As the PKG generates and holds all secret keys for all users, a complete trust must be placed on the PKG. However, it is difficult to ensure complete trust in the real world scenario. For example, a malicious PKG can sell users' keys to get profit or even simulate any user to sign messages. This is known as the key escrow problem and it seems to be inherent in IBC. Thus, Boneh and Frankhn [5] utilized secret sharing scheme with multiple PKGs to resolve the key escrow problem, in which the master secret key is jointly computed by multiple PKGs, such that no single PKG has the knowledge of it. But this method needs an extra computation and communication overhead between multiple PKGs and the users.

In order to provide a solution, Al-Riyami and Paterson [3] introduced the notion of Certificateless Cryptography (CLC) which eliminates the use of certificates in PKC and solve the key escrow problem in IBC. The first component is chosen by the user as his/her secret value, and it is not known to PKG. On the other hand, the second component is the partial private key, which is generated by PKG. So the full private key of each user is composed with two components. In addition, CLC is not identity-based because the user has an additional random public key. Even now, some existing certificateless schemes [10, 15, 29] are showed to be insecure, while others secure schemes [9, 30] and [18, 19] have low efficiencies. Therefore it is not easy to construct a secure and efficient certificateless scheme.

Obviously, the identity-based signature scheme without key escrow can avoid some security risks. For example, it can avoid malicious PKG or adversary by obtaining user's private key to do some illegal actions. In addition, if PKG forges the secret value of a user, and binds his/her ID by more than two private keys to one adversary. That also brings security risks to the scheme.

Because of its importance and wide applicability, efficient and provably secure identity-based online/offline signature scheme is becoming the research focus. However, the most existing identity-based online/offline signature schemes use bilinear pairing and a probabilistic MTP function [13, 25, 26, 27, 28]. The relative computational cost of bilinear pairing is approximately two to three times more than the elliptic curve point multiplication. The MTP function also needs more execution time than an elliptic curve point multiplication [11, 12]. Therefore, how to design an efficient and secure identity-based online/offline signature scheme without using bilinear pairing and MTP function is still an important and practical issue.

Motivated by these concerns, in this paper, we construct an efficient identity-based online/offline signature scheme without using bilinear pairing and MTP function, which avoids the key escrow problem by adopting the idea of CLC. PKG only produces a partial private key while the user generates the other partial private key, which both them are the full private key. So the PKG does not have full knowledge of the user's private key. Further, user's public key can be directly related with his/her ID in our scheme, which is different from those schemes of CLC. It shows clearly that our scheme is more secure and efficient than some certificateless schemes [9, 18, 19]. In our scheme, even if PKG forges the signature for the user, the user can generate evidence and submit to the intercessor trusted authority (TA). According to the evidence, TA can check whether the PKG is honest. The proposed scheme is provably secure and computationally more efficient than existing schemes. Our scheme also provides security proof under random oracle model with the difficulties of ECDLP and against adaptive chosen ID and message attacks.

The rest of this paper is organized as follows. We review some preliminary works in Section 2, and Section 3 describes the syntax of ID-based online/offline signature. In Section 4, we construct an identity-based online/offline signature scheme without key escrow problem. The security of our scheme is given in Section 5. We then give performance evaluation in Section 6. Finally, concluding remarks are given in Section 7.

2 Preliminary Works

In this section, we review certain related definitions and mathematical problem, which will be introduced in our proposed scheme.

Definition 1 (**Elliptic Curve Over Z_p**). *Let $p > 3$ be an odd prime. An elliptic curve E over Z_p is defined by an equation of the from $y^2 = x^3 + ax + b$, where $x, y, a, b \in Z_p$ and $(4a^3 + 27b^2) \bmod p \neq 0$. Then we define $G_p = \{(x, y) : x, y \in Z_p and (x, y) \in E(Z_p)\} \cup \{O\}$ as the additive elliptic curve group, and the point O is known as point at infinity or zero point.*

Definition 2 (**Elliptic Curve Discrete Logarithm Problem (ECDLP)**). *Given two group elements $P \in G_1$ and $Q \in G_1$, to find an integer $a \in Z_q^*$, such that $Q = aP$ whenever such an integer exists.*

Lemma 1 (**Forking Lemma [17]**). *We assume that A be a attacker within a time bound t_1, A produces a valid signature $(m, \sigma_1, h, \sigma_2)$ with probability $\varepsilon \geq 10(S+1)(S+H)/2^k$. If the triples (σ_1, h, σ_2) can be simulated without knowing the secret key, with an indistinguishable distribution probability, then there is another machine which has control over the machine obtained from A replacing interaction with the signer by simulation and produces two valid signatures $(m, \sigma_1, h, \sigma_2)$ and $(m, \sigma_1, h', \sigma_2)$ such that $h \neq h'$. We denote respectively by S and H the number of queries that A can ask to the signer and the number of queries that A can ask to the random oracle.*

3 The Syntax of ID-based Online/Offline Signature

In the section, we briefly describe the syntax of ID-based online/offline signature.

An ID-based online/offline signature scheme is composed mainly of the following five polynomial algorithms.

1) **IO_Setup:** The parameter generation is a probability algorithm running by PKG, in which it inputs a security parameter 1^k, and then outputs system parameters *params* and a master key IO_Msk.

2) **IO _Etract:** The extract algorithm is a key generation algorithm, in which it inputs the system parameters *params*, an identity ID, and a master key IO_Msk, and accordingly returns the private key IO_sk_{ID} the user.

3) **IO _OffSign:** The online signing algorithm is a probability algorithm that by inputting the system parameters *params*, and user's private key IO_sk_{ID}, the user calculates the offline signature S and stores it.

4) **IO_OnSign:** The offline signing algorithm is a probability algorithm that after inputting a message m and offline signature S, it outputs an online signature σ.

5) **IO_Verify:** The verification algorithm is a deterministic algorithm that by inputting the message m, identity ID, offline signature S, and online signature σ, finally it outputs either accept or reject.

4 The Proposed Scheme

In this section, we proposed an identity-based online/offline signature scheme without Key Escrow. Similarly, our scheme is also comprised of five polynomial time algorithms: **IO_Setup**, **IO_Extract**, **IO_OffSign**, **IO_OnSign** and **IO_Verify**. In addition, the related details of our scheme are shown in Figure 1 and Figure 2.

1) **IO_Setup:**

Let $p > 3$ be an odd prime. An elliptic curve E over Z_p is defined by an equation of the from $y^2 = x^3 + ax + b$, where $x, y, a, b \in Z_p$ and $(4a^3 + 27b^2)$ mod $p \neq 0$. Then it runs the parameter generator by inputting a security parameter $\lambda \in Z^+$ to generate a prime $q > 2^\lambda$, and choose a group G of prime order q. Suppose that P be the basic point of E, where the prime order of P is q. The group $G =< P >$ is based on the Discrete Logarithm problem. Then PKG picks a master key $s \in_R Z_q^*$ randomly and computes

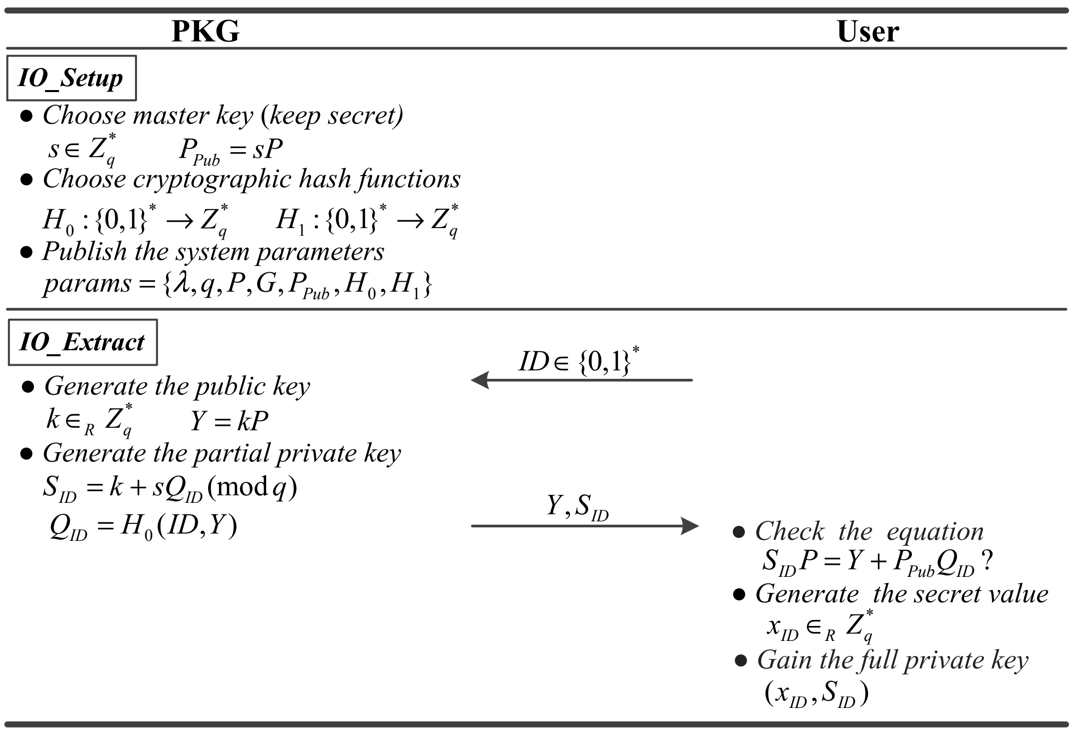
$$P_{Pub} = sP. \tag{1}$$

Furthermore, PKG chooses two cryptographic hash functions $H_0 : \{0, 1\}^* \rightarrow Z_q^*$ and $H_1 : \{0, 1\}^* \rightarrow Z_q^*$. Finally, PKG publishes the system parameter $params = \{\lambda, q, P, G, P_{Pub}, H_0, H_1\}$ and keeps the master key s in secret.

2) **IO_Etract:**

We split this extract process into two phases: the one is to generate the partial private key; the other is to generate the secret value. Given an identity ID, PKG chooses $k \in_R Z_q^*$ randomly and computes

$$\begin{aligned} Y &= kP, \\ S_{ID} &= k + sQ_{ID}(\bmod q). \end{aligned}$$



PKG **User**

IO_Setup

- *Choose master key (keep secret)*
  $s \in Z_q^*$ $\quad P_{Pub} = sP$
- *Choose cryptographic hash functions*
  $H_0 : \{0,1\}^* \rightarrow Z_q^*$ $\quad H_1 : \{0,1\}^* \rightarrow Z_q^*$
- *Publish the system parameters*
  $params = \{\lambda, q, P, G, P_{Pub}, H_0, H_1\}$

IO_Extract

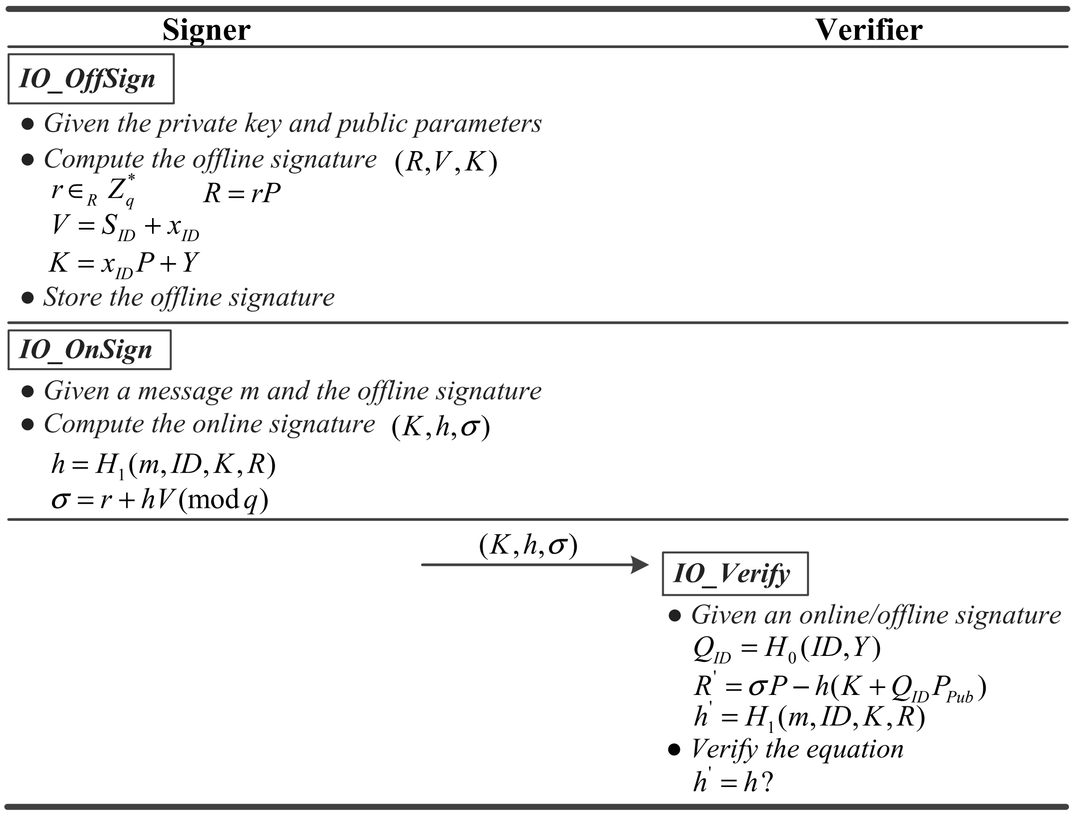
$\xleftarrow{ID \in \{0,1\}^*}$

- *Generate the public key*
  $k \in_R Z_q^*$ $\quad Y = kP$
- *Generate the partial private key*
  $S_{ID} = k + sQ_{ID} (\bmod q)$
  $Q_{ID} = H_0(ID, Y)$

$\xrightarrow{Y, S_{ID}}$

- *Check the equation*
  $S_{ID}P = Y + P_{Pub}Q_{ID}$?
- *Generate the secret value*
  $x_{ID} \in_R Z_q^*$
- *Gain the full private key*
  (x_{ID}, S_{ID})

Figure 1: IO_Setup and $IO\_Extract$ phases of the proposed scheme



Signer **Verifier**

IO_OffSign

- *Given the private key and public parameters*
- *Compute the offline signature* (R, V, K)
  $r \in_R Z_q^*$ $\quad R = rP$
  $V = S_{ID} + x_{ID}$
  $K = x_{ID}P + Y$
- *Store the offline signature*

IO_OnSign

- *Given a message m and the offline signature*
- *Compute the online signature* (K, h, σ)
  $h = H_1(m, ID, K, R)$
  $\sigma = r + hV (\bmod q)$

$\xrightarrow{(K, h, \sigma)}$

IO_Verify

- *Given an online/offline signature*
  $Q_{ID} = H_0(ID, Y)$
  $R' = \sigma P - h(K + Q_{ID}P_{Pub})$
  $h' = H_1(m, ID, K, R)$
- *Verify the equation*
  $h' = h$?

Figure 2: $IO\_OffSign$, IO_OnSign and IO_Verify phases of the proposed scheme

PKG takes S_{ID} as the partial private key, and sends S_{ID} and Y to the user, where $Q_{ID} = H_0(ID, Y)$. The user may check whether the following equation holds or not

$$S_{ID}P = Y + P_{Pub}Q_{ID}. \tag{2}$$

If the equation holds, the user confirms that his partial private key is valid.

Then the user picks $x_{ID} \in_R Z_q^*$ randomly and sets x_{ID} as the secret value.

Finally the full private key of the user is (x_{ID}, S_{ID}).

3) **IO_OffSign:**

Given the private key (x_{ID}, S_{ID}), the signer picks $r \in_R Z_q^*$ randomly and computes the offline signature triple (R, V, K).

$$\begin{aligned} R &= rP, \\ V &= S_{ID} + x_{ID}, \\ K &= x_{ID}P + Y. \end{aligned}$$

Please note that these offline components may be computed when the device is idle, because these components are independent of the messages.

4) **IO_OnSign:**

Given a message m and the offline signature triple (R, V, K), the signer computes the online signature

$$\begin{aligned} h &= H_1(m, ID, K, R), \\ \sigma &= r + hV(\bmod q). \end{aligned}$$

Finally, the full signature is the triple (K, h, σ).

5) **IO_Verify:**

Given an online/offline signature triple (K, h, σ) on the message m for an identity ID, the verifier computes

$$\begin{aligned} Q_{ID} &= H_0(ID, Y), \\ R' &= \sigma P - h(K + Q_{ID}P_{Pub}), \quad (3) \\ h' &= H_1(m, ID, K, R). \quad (4) \end{aligned}$$

Then he checks whether the following equation holds or not,

$$h' = h. \tag{5}$$

If the equation holds, the signature is valid; otherwise it is invalid.

Correction: We prove that the proposed scheme is correct as follows:

$$\begin{aligned} R' &= \sigma P - h(K + Q_{ID}P_{Pub}) \\ &= (r + hV)P - h(K + Q_{ID}P_{Pub}) \\ &= (r+h(S_{ID}+x_{ID}))P - h(K+Q_{ID}P_{Pub}) \\ &= (r+h(S_{ID}+x_{ID}))P - h(x_{ID}P+Y+Q_{ID}P_{Pub}) \\ &= (r+h(k+sQ_{ID}+x_{ID}))P - h(x_{ID}P+kP+Q_{ID}P_{Pub}) \\ &= (r+h(k+sQ_{ID}+x_{ID}))P - h(x_{ID}P+kP+sQ_{ID}P) \\ &= (r+h(k+sQ_{ID}+x_{ID}))P - h(x_{ID}+k+sQ_{ID})P \\ &= rP \\ &= R. \end{aligned}$$

By $R' = R$, and Equation (4), we can easily get the verification equation (i.e. Equation (5)) is correct.

5 Security Proof

In this section, we show that our proposed scheme is secure.

The security proof of our scheme satisfies existentially unforgeable under adaptive chosen message attacks and ID attacks.

If PKG wants to simulate a user to sign a message, then the types of his behaviors may be as follows:

1) The PKG can forge user's signature without knowing the secret value and the partial private key of the target user.

2) The PKG can forge user's signature that has the knowledge of the master key or the partial private key, but does not have the secret value of the target user.

3) The PKG can forge user's signature that has the knowledge of the secret value and the partial private key of the target user. (But here, the secret value is replaced by PKG, rather than the truly secret value.)

According to the behaviors that describe above, we define the corresponding adversary into three classes. And then shows the following proofs that our scheme is secure.

Theorem 1. *There exists a adversary A_1 who can satisfies existential unforgeability against adaptive chosen ID and message attacks without knowing the user's secret value and the partial private key in random oracle model under the ECDLP assumption.*

Proof. Let A_1 be an adversary who can break our improved scheme. We show how A_1 can be used by a Probabilistic Polynomial Time (PPT) algorithm C to solve the ECDLP. □

Suppose that (P, aP) as a random ECDLP stance of a group G and outputs a. Algorithm C will do the following simulations by interacting with A_1.

Setup: In this stage, Algorithm C performs as follows.

1) Algorithm C sets the public key $P_{Pub} = aP$ and publishes the system parameters $params = \{\lambda, q, P, G, P_{Pub}, H_0, H_1\}$.
2) For $1 \leq u \leq q_{H_0}$, Algorithm C chooses ID_u randomly as the challenge identity for this game, where q_{H_0} denotes the maximum number of querying H_0 oracle.
3) Algorithm C chooses $Q_u \in_R {Z_q}^*$ randomly, sets $Y_u = -Q_u(aP)$, defines $H_0(ID_u, Y_u) = Q_u$, and then adds (ID_u, Y_u, Q_u) on the ${H_0}^{list}$.
4) Algorithm C gives A_1 system parameters $params = \{\lambda, q, P, G, P_{Pub}, H_0, H_1\}$.

Then C starts by answering queries from A_1 as follows.

H_0 **Queries:** A_1 inputs (ID_i, Y_i), and Algorithm C calls the ${H_0}^{list}$ list. If the list ${H_0}^{list}$ contains (ID_i, Y_i, Q_i), C returns to A. Otherwise, C chooses $Q_i \in_R {Z_q}^*$ randomly, adds (ID_i, Y_i, Q_i) to the list ${H_0}^{list}$, and returns Q_i to A.

H_1 **Queries:** A_1 inputs (m_i, ID_i, K_i, R_i), and Algorithm C calls the ${H_1}^{list}$ list. C scans the list ${H_1}^{list}$ to check whether has already been defined. Otherwise, C picks $h_i \in_R {Z_q}^*$ randomly, adds $(m_i, ID_i, K_i, R_i, h_i)$ to the list, and returns h_i to A_1.

Key Extract Queries: We split this query into two phases: the secret extract value and the partial private key extract queries.

Partial private key extract queries: when A_1 requests the private key associated with the identity ID_i, C checks whether the equation of $ID_i = ID_u$ holds or not and maintains the E^{list} list.

1) If $ID_i = ID_u$, then C halts and outputs "failure".
2) If $ID_i \neq ID_u$, C picks $x_{ID_i} \in_R {Z_q}^*$ randomly as the secret value associated with the identity ID_i. Then C chooses $S_{ID_i} \in_R {Z_q}^*$ and computes $K_i = S_{ID_i}P + x_{ID_i}P - Q_i aP$, where $Q_i = H_0(ID_i, Y_i)$. If $H_0(ID_i, Y_i, Q_i)$ has already been defined, then C halts and outputs "failure" (denote the event by E_1). C adds (ID_i, Y_i, Q_i) and $(ID_i, x_{ID_i}, S_{ID_i})$ to the list.. and E^{list}, respectively. Finally, C returns K_i, S_{ID_i}.The probability of the event E_1 is at most $(q_{H_0} + q_E)/2^{\lambda+1}$, where q_E denotes the number of querying key extraction oracle.

Secret value extract queries: If $ID_i = ID_u$, then C halts and outputs "failure"; otherwise finds $(ID_i, x_{ID_i}, S_{ID_i})$ from the list E^{list} and returns associated secret value x_{ID}.

Signing Queries: Assume queries a signature for an identity ID and a message m.

1) If $ID_i = ID_u$, then C chooses $\sigma_u, h_u \in_R {Z_q}^*$ randomly, sets $K_u = aP - Q_u(aP)$ and computes $R_u = \sigma_u P - h_u(K_u + Q_u P_{Pub})$, where $h_u = H_1(m_i, ID_u, K_u, R_u)$. If $H_1(m_i, ID_u, K_u, R_u)$ has already been defined, then C halts and outputs "failure" (denote the event by E_2). Finally, C returns (K_u, h_u, σ_u) as a signature. The probability of the event E_2 is at most $(q_{H_1} + q_S)/2^{\lambda}$, where q_s denotes the maximum number of querying signing oracle.
2) If $ID_i \neq ID_u$, the signature is ordinary, because of C has the secret value and the partial private key. That is, to say C can perform online signing algorithm normally and generate online signature accordingly.

Forgery: Suppose that A_1 outputs a forgeable signature (K^*, h^*, σ^*) on a message m^* for an identity ID^*. Here, ID^* is not submitted to partial private key extract oracle and secret value extract oracle, and (m^*, ID^*) is not query to signing oracle.

1) If $ID^* \neq ID_u^*$ and $K^* \neq K_u^*$, then C halts and outputs "failure" (denote the event by E_3). The probability of the event E_3 is not less than $1/q_{H_0}$.
2) Otherwise, according to forking lemma, there exists an algorithm B which generates two valid signatures $(m^*, ID_u, K_u, R, h_1, \sigma_1)$ and $(m^*, ID_u, K_u, R, h_2, \sigma_2)$ in PPT, where $h_1 \neq h_2$ and Q_u is steadiness due to $H_0(ID_u, Y_u) = Q_u$. So the equations hold as follows:

$$R = \sigma_1 P - h_1(K_u + Q_u P_{Pub}) \; by \; Equation \; (3)$$

$$R = \sigma_2 P - h_2(K_u + Q_u P_{Pub}) \; by \; Equation \; (3).$$

After the division, we get $(\sigma_1 - \sigma_2)P = (h_1 - h_2)aP$, then obtain $a = (\sigma_1 - \sigma_2)/(h_1 - h_2)$, and return a as the solution to the ECDLP instance.

Theorem 2. *There exists an adversary who can satisfy existential unforgeability on adaptively chosen ID and message attacks. And has the knowledge of the master key or partial private key, but does not have the user's secret value in random oracle model under the ECDLP assumption.*

Proof. Let A_2 be an adversary who can break our improved scheme. We show how A_2 can be used by a PPT algorithm C to solve the ECDLP. □

Suppose that (P, aP) as a random ECDLP stance of a group G and outputs a. Algorithm C will do the following simulations by interacting with A_2.

Setup: In this stage, Algorithm C performs as follows.

1) Algorithm C sets the public key $P_{Pub} = aP$ and publishes the system parameters, $params = \{\lambda, q, P, G, P_{Pub}, H_0, H_1\}$.
2) For $1 \leq u \leq q_{H_0}$, Algorithm C chooses ID_u randomly as the challenge identity for this game, where q_{H_0} denotes the maximum number of querying H_0 oracle.
3) Algorithm C gives system parameters $params = \{\lambda, q, P, G, P_{Pub}, H_0, H_1\}$ and the master key s.

Then C starts by answering queries from A_2 as follows.

H_0 **Queries:** A_2 inputs (ID_i, Y_i), Algorithm C calls the ${H_0}^{list}$ list. If the list ${H_0}^{list}$ contains (ID_i, Y_i, Q_i), C returns Q_i to A. Otherwise, C chooses $Q_i {\in}_R {Z_q}^*$ randomly, adds (ID_i, Y_i, Q_i) to the list ${H_0}^{list}$ and returns Q_i to A_2.

H_1 **Queries:** A_2 inputs (m_i, ID_i, K_i, R_i), and Algorithm C calls the ${H_1}^{list}$ list. C scans the list to check whether has already been defined. Otherwise, C picks $h_i {\in}_R {Z_q}^*$ randomly, adds $(m_i, ID_i, K_i, R_i, h_i)$ to the list ${H_1}^{list}$, and returns h_i to A_2.

Key Extract Queries: We split this query into two phases: the secret value extract and the partial private key extract queries. Since A_2 only knows user's partial private key without knowing the secret key, this phase can leave out the secret value queries.

Partial private key extract queries: when A_2 requests the private key associated with identity ID_i, C checks whether $ID_i = ID_u$ holds or not and maintains the E^{list} list.

1) If $ID_i = ID_u$, then C sets $Y_i = aP$ and finds (ID_i, Y_i, Q_i) from the ${H_0}^{list}$ list. C chooses $k_i {\in}_R {Z_q}^*$ randomly and computes $S_{ID_i} = k_i + sQ_i$, then adds $(ID_i, S_{ID_i}, \perp)$ to the list (ID_i, S_{ID_i}, k_{i1}) ($\perp$ denotes the unknown secret value for ID_i). Finally, returns S_{ID_i}.
2) If $ID_i \neq ID_u$, C finds (ID_i, Y_i, Q_i) from the ${H_0}^{list}$ list. Then C picks $k_{i1}, k_{i2} {\in}_R {Z_q}^*$ randomly and computes $S_{ID_i} = k_{i2} + sQ_i$. Add (ID_i, S_{ID_i}, k_{i1}) to the K_i list. Finally, C returns S_{ID_i}.

Signing Queries: Assume A_2 queries a signature for an identity ID and a message m.

1) If $ID_i = ID_u$, then C chooses $\sigma_i, h_i {\in}_R {Z_q}^*$ randomly, sets $Y_i = aP$, and finds (ID_i, Y_i, Q_i) from the list ${H_0}^{list}$, and further C sets $K_i = Y_i = aP$ and computes $R_i = \sigma_i P - h_i(K_i + Q_i P_{Pub})$, where $h_i = H_1(m_i, ID_i, K_i, R_i)$. If $H_1(m_i, ID_i, K_i, R_i)$ has already been defined, then C halts and outputs "failure" (denote the event by E_2). Finally, C returns (K_i, h_i, σ_i) as an online signature. The probability of the event E_2 is at most $(q_{H_1} + q_S)/2^\lambda$, where q_S denotes the maximum number of querying signing oracle.
2) If $ID_i \neq ID_u$, the signature is ordinary, because of C has the secret value and the partial private key. That's to say C can perform online signing algorithm normally and generate online signature accordingly.

Forgery: Suppose that A_2 outputs a forgeable signature (K^*, h^*, σ^*) on a message m^* for an identity ID^*. Here, ID^* is not submitted to secret value extract oracle, and (m^*, ID^*) is not query to signing oracle.

1) If $ID^* \neq ID_u^*$ and $K^* \neq K_u^*$, then C halts and outputs "failure" (denote the event by E_3). The probability of the event E_3 is not less than $1/q_{H_0}$.
2) Otherwise, according to forking lemma, there exists an algorithm B generates two valid signatures $(m^*, ID_u, K_u, R, h_1, \sigma_1)$ and $(m^*, ID_u, K_u, R, h_2, \sigma_2)$ in PPT, where $h_1 \neq h_2$ and $Y' = K'P$ is steadiness. So the equations hold as follows:

$$R = \sigma_1 P - h_1(K_u + Q_u P_{Pub}) \text{ by Equation } (3)$$

$$R = \sigma_2 P - h_2(K_u + Q_u P_{Pub}) \text{ by Equation } (3).$$

After the division, we get $(\sigma_1 - \sigma_2)P = (h_1 - h_2)(a + sQ_u)P$, then obtain $a = (\sigma_1 - \sigma_2)/(h_1 - h_2) - sQ_u$, and return a as the solution to the ECDLP instance.

Theorem 3. *If PKG simulates a legitimate user to forge the signature, who has the knowledge of the user's secret value and the partial private key (the secret value is not real, which represents an alternative), then we can prove to the intercessor that PKG mentioned above is dishonest.*

Proof. According to the Theorem 1 and Theorem 2, our scheme is unforgeable for the honest PKG or the negative dishonest PKG. We split this process into two steps: to forge private key and sign message. □

1) Forge Private Key.

   Assume that ID as user's identity and (x_{ID}, S_{ID}) as the private key. Then PKG simulates the user to sign messages by two ways:

   a. Knowing the user's secret value x_{ID};
   b. Replacing the user's secret value x_{ID}. However, the user chooses x_{ID} randomly and thus it is impossible for PKG to get x_{ID}.

   Thus, PKG only chooses to replace the user's secret value x_{ID} and generates another private key. The details steps as follows:

Table 1: Notions and definitions of time complexities

Notions	Definitions
T_M	Time required for executing a modular multiplication operation.
T_{ZM}	Time required for executing a multiplication operation in Z_q^*.
T_{GM}	Time required for executing a multiplication operation in group.
T_{GE}	Time required for executing an exponentiation operation in group.
T_H	Time required for executing a hash operation.
T_E	Time required for executing a modular exponentiation operation, $T_E \approx 240T_M$.
T_P	Time required for executing a bilinear pairing operation,$T_P \approx 87T_M$.
T_{PE}	Time required for executing an pairing-based exponentiation operation, $T_{PE} \approx 43.5T_M$.
T_{PM}	Time required for executing an elliptic curve scalar point multiplication operation, $T_{PM} \approx 29T_M$.
T_{MTP}	Time required for executing a hash function operation, $T_{MTP} \approx T_{PM} \approx 29T_M$.
T_{PA}	Time required for executing a point addition operation of two elliptic curve points, $T_{PA} \approx 0.12T_M$.

a. PKG chooses x_{ID} to replace the user's secret value (the probability of $x_{ID'} = x_{ID}$ can $S_{ID'} = k' + sQ_{ID'}(\text{mod}q)$ be ignored).

b. PKG picks $k' \in_R {Z_q}^*$ randomly and computes $Y' = k'P$ and $S'_{ID} = k' + sQ'_{ID}\,(\text{mod}q)$, where $Q_{ID'} = H_0(ID, Y')$. Assume that Y', S'_{ID} satisfy the Equation (2), then it outputs the private key (x'_{ID}, S'_{ID}).

2) Sign Message.

After PKG forging the user's private key $(x_{ID'}, S_{ID'})$, he starts to perform signing algorithm. (k', h', σ') denotes the signature on a message m for the user. The user can execute signing algorithm twice to prove whether (k', h', σ') is forged by PKG or an adversary colluded with PKG. Suppose that the user generates two signatures (K, h_1, σ_1) and (K, h_2, σ_2), and submits (K, h_1, σ_1) and (K, h_2, σ_2) to the intercessor TA.

But here $K' \neq K$. If PKG wants to make $K' = K$, then PKG needs to hold the equation $(k' + x'_{ID})P = (k + x_{ID})P$. Further PKG requires to know the value $Y' = (k + x_{ID} - x'_{ID})P = k'P$, but PKG does not know x_{ID}. According to the ECDLP, PKG can't obtain k, Q_{ID} and S_{ID}, so $K' \neq K$.

If three signatures above are valid, then K in (K, h_1, σ_1) and (K, h_2, σ_2) is same. We get $K' \neq K$ in (K', h', σ'), so (K', h', σ') must be forged by PKG or an adversary colluded with PKG.

In summary, the Theorem 1 and Theorem 2 can make our scheme unforgeable for the honest PKG or the negative dishonest PKG. Only PKG knows the master key s, and thus only PKG can replace the user's secret value to generate another private key (x'_{ID}, S'_{ID}). According to the above, PKG can not make $K' = K$, so we can prove whether PKG is honest or not by checking the equation $K' = K$ or not.

6 Performance Evaluation

In this section, we evaluated the performance of our scheme and gave a detailed comparison with other schemes proposed in the literatures [9, 13, 18, 19, 26, 27, 28] proposed in the literature. To estimate the operating overhead, we define the notations in Table 1. Please note that the time of other light operations is ignored in the comparisons (e.g., addition operation in Z_q^*), since it is relatively smaller.

In most online/offline signatures, the main computational costs are shifted to the offline phase, so the efficiency is dependent on the online and verification phase. The comparisons of our scheme with other online/offline schemes [9, 13, 18, 19, 26, 27, 28] are listed in Table 2. It is obvious that our scheme does not require any bilinear pairing and MTP hush function operations. Therefore, it is more efficient than these schemes [9, 13, 18, 24, 28] in terms of computational costs in the online signing, and also more efficient than other schemes [9, 13, 18, 19, 26, 27, 28] in terms of computational costs in the verification.

In Table 3[1], the comparisons for different aspects between our scheme and other related schemes are summarized. This table shows that our scheme supports all these good attributes, but the schemes proposed in [9, 13, 18, 19, 26, 27, 28] do not.

To sum up, our identity-based online/offline signature scheme enjoys the following good advantages: (1) No bilinear pairing and probabilistic MTP function, (2) Low computation costs, (3) No key escrow problem, (4) Key confirmation, (5) Provable security under the random oracle model against adaptive chosen ID and message attacks, and (6) supports traceability.

[1]Note: A_1, -based cryptosystem; A_2, Hardness problems; A_3, Random oracle; A_4, No bilinear pairing and MTP hash function; A_5, Low computational costs; A_6, No key escrow problem; A_7, Key confirmation; A_8, Provable security; A_9, Traceability; IBC, Identity-Based Cryptosystem; CLC, Certificateless Cryptosystem; CDHP, Computation Diffie-Hellman Problem; GDH, Gap Diffie-Hellman Problem; IFP, Integer Factorization Problem; ECDLP, Elliptic Curve Discrete Logarithm Problem.

Table 2: Computational cost comparison of the proposed scheme with others

Schemes	**The online signing phase**	**The verification phase**	**The total computational cost**
Xu et al. [27]	$1T_{ZM} + 1T_H$	$2T_{PM} + 2T_P + 1T_{MTP}$ $+1T_{PA} + 1T_H$ $\approx 261.12T_M + 1T_H$	$\approx 261.12T_M + 1T_{ZM} + 2T_H$
Xu et al. [26]	$1T_{ZM} + 1T_H$	$2T_{PM} + 2T_P + 1T_{MTP}$ $+2T_{PA} + 1T_H$ $\approx 261.24T_M + 1T_H$	$\approx 261.24T_M + 1T_{ZM} + 2T_H$
Ge et al. [9]	$2T_E + 2T_M + 1T_H$ $\approx 482T_M + 1T_H$	$4T_E + 2T_M + 4T_{GM} + 3T_H$ $\approx 962T_M + 4T_{GM} + 3T_H$	$\approx 1444T_M + 4T_{GM} + 4T_H$
Wu et al. [25]	$1T_{ZM} + 1T_H$	$2T_{PM} + 2T_P + 1T_{MTP}$ $+2T_{PA} + 1T_H$ $\approx 261.24T_M + 1T_H$	$\approx 261.24T_M + 1T_{ZM} + 2T_H$
Ming et al. [28]	$2T_{ZM} + 1T_H$	$3T_{PM} + 2T_P + 1T_{MTP}$ $+1T_{PA} + 2T_H$ $\approx 290.12T_M + 2T_H$	$\approx 290.12T_M + 2T_{ZM}+3T_H$
Selvi et al. [18]	$2T_{ZM} + 2T_H$	$3T_{GM} + 4T_{GE} + 4T_H$	$\approx 3T_{GM} + 3T_{GE}$ $+2T_{ZM} + 6T_H$
Selvi et al. [19]	$1T_{ZM} + 1T_H$	$6T_{PM} + 4T_{PA} + 4T_P$ $+3T_{MTP} + 2T_{ZM} + 3T_H$ $\approx 609.48T_M + 2T_{ZM}$	$\approx 609.48T_M + 3T_{ZM} + 4T_H$
Wang et al. [24]	$(n-1)T_{ZM}$	$3T_E + 2T_{ZM}$ $\approx 720T_M + 2T_{ZM}$	$\approx 720T_M + (n+1)T_{ZM}$
Kar et al. [13]	$2T_{PM} + 2nT_{ZM} + 1T_H$ $\approx 58T_M + 2nT_{ZM} + 1T_H$	$3T_P + 1T_{PE} + 1T_{GM}$ $+1T_{MTP} + 1T_H$ $\approx 304.5T_M + 1T_{GM}$	$\approx 362.62T_M + 1T_{GM}$ $+2nT_{ZM} + 2T_H$
Beak et al. [4]	$(n-1)T_{ZM} + 1T_{GE}$ $+1T_H + 1T_M$	$3T_{GE} + 3T_{GM} + 1T_H$	$(n-1)T_{ZM} + 4T_{GE} + 3T_{GM}$ $+2T_H$
Abinav et al. [1]	$1T_{ZM} + 1T_H$	$6T_{PM} + 4T_{PA} + 3T_H$ $+1T_{MTP}$	$\approx 203.48T_M + 4T_H + 1T_{ZM}$
Aboud et al. [2]	$(n-1)T_{ZM} + 1T_H + 1T_M$	$3T_P + 1T_{PE} + 1T_{GM}$ $+1T_{MTP} + 1T_H$ $\approx 304.5T_M + 1T_{GM}$	$\approx 3T_{GE} + (n+1)T_{ZM}$ $+2T_H + 1T_M$
Our Scheme	$1T_{ZM} + 1T_H$	$3T_{GE} + 2T_{ZM} + 1T_H + 1T_M$	$\approx 87.24T_M + 1T_{ZM} + 3T_H$

Table 3: Comparisons of the proposed scheme with the existing schemes for different attributes

Schemes	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9
Xu et al. [27]	IBC	CDHP	Yes	No	No	No	No	No	No
Xu et al. [26]	IBC	CDHP	Yes	No	No	No	No	No	No
Ge et al. [9]	CLC	ECDLP	Yes	Yes	No	Yes	Yes	Yes	No
Wu et al. [25]	IBC	CDHP	Yes	No	No	No	No	Yes	No
Ming et al. [28]	IBC	CDHP	Yes	No	No	No	No	Yes	No
Selvi et al. [18]	CLC	GDHP	No	Yes	No	Yes	Yes	No	No
Selvi et al. [19]	CLC	GDHP	Yes	No	No	Yes	Yes	Yes	No
Wang et al. [24]	IBC	IFP	No	Yes	No	No	No	No	No
Kar et al. [13]	IBC	CDHP	Yes	No	No	No	No	Yes	No
Beak et al. [4]	IBC	ECDLP	Yes	Yes	No	No	Yes	Yes	No
Abinav et al. [1]	CLC	CDH	Yes	No	No	Yes	Yes	Yes	No
Aboud et al. [2]	IBC	ECDLP	Yes	Yes	No	Yes	Yes	Yes	No
Our Scheme	IBC	ECDLP	Yes	Yes	Yes	Yes	Yes	Yes	Yes

7 Conclusion

ID-based online/offline signature scheme is a combination of IBC and online/offline signature. It has the following advantages: (1) eliminating the costly certificate verification process and the storage of the length certificate, (2) producing some computation results in the offline phase which is stored in advance and later used when the message to be signed is known, such that a valid signature can be generated quickly in the online phase. In this paper, we designed an bilinear paring and MTP function-free identity-based online/offline signature scheme without employing complex bilinear paring and MTP function, which was proven to satisfy the existential unforgeablility against adaptively chosen message and ID attacks in the random oracle under the ECDLP assumption. Our proposed scheme is computationally more efficient than other related signature schemes. Especially, our scheme provides Key confirmation and Traceability. Next, applying the proposed scheme in some resource-constrained environments is our future work.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Nos 61173187, 11301002 and 61572001), Natural Science Foundation of Anhui Province (No 1408085QF107), the 211 Project of Anhui University (Nos 33190187 and 17110099), and the Educational Commission of Anhui Province, China (KJ2015A326).

References

[1] K. Abinav, S. Badrinarayanan, C. P. Rangan, S. S. D. Selvi, S. S. Vivek, and V. K. Pradhan, "A revocable online-offline certificateless signature scheme without pairing," *IACR Cryptology ePrint Archive*, vol. 2013, pp. 758, 2013.

[2] S. J. Aboud, "Secure online-offline identity-typed signature scheme," *International Journal of Scientific Research and Management Studies*, vol. 1, pp. 2349–3371, 2015.

[3] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Advances in cryptology (ASIACRYPT'03)*, pp. 452–473, 2003.

[4] J. Baek, Y. J. Byon, E. Hableel, and M. Al-Qutayri, "An authentication framework for automatic dependent surveillance-broadcast based on online/offline identity-based signature," in *2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC'13)*, pp. 358–363, 2013.

[5] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology (CRYPTO'01)*, pp. 213–229, 2001.

[6] X. Chen, F. Zhang, W. Susilo, and Y. Mu, "Efficient generic on-line/off-line signatures without key exposure," in *Applied Cryptography and Network Security*, pp. 18–30, 2007.

[7] Y. Desmedt and J. J. Quisquater, "Public-key systems based on the difficulty of tampering (is there a difference between DES and RSA)," in *Advances in Cryptology (CRYPTO'86)*, pp. 111–117, 1986.

[8] S. Even, O. Goldreich, and S. Micali, "Online/offline digital signatures," in *Advances in Cryptology (CRYPTO'90)*, pp. 263–375, 1990.

[9] A. Ge, S. Chen, and X. Huang, "A concrete certificateless signature scheme without pairings," in *MInternational Conference on ultimedia Information Networking and Security (MINES'09)*, vol. 2, pp. 374–377, 2009.

[10] M. C. Gorantla and A. Saxena, "An efficient certificateless signature scheme,". in *Computational Intelligence and Security*, pp. 110–116. 2005.

[11] S. K. H. Islam and G. P. Biswas, "A pairing-free identity-based authenticated group key agreement protocol for imbalanced mobile networks," *Annals of Telecommunications*, vol. 67, no. 11-12, pp. 547–558, 2012.

[12] S. K. H. Islam and G. P. Biswas, "Provably secure and pairing-free certificateless digital signature scheme using elliptic curve cryptography," *International Journal of Computer Mathematics*, vol. 90, no. 11, pp. 2244–2258, 2013.

[13] J. Kar, "Provably secure online/off-line identity-based signature scheme for wireless sensor network," *International Journal of Network Security*, vol. 16, no. 1, pp. 29–39, 2014.

[14] A. V. N. Krishna, A. H. Narayana, K. M. Vani, "Window method based cubic spline curve public key cryptography," *International Journal of Electronics and Information Engineering*, vol. 4, no. 2, pp. 94–102, 2016.

[15] X. X. Li, K. F. Chen, and L. Sun, "Certificateless signature and proxy signature schemes from bilinear pairings," *Lithuanian Mathematical Journal*, vol. 45, no. 1, pp. 76–83, 2005.

[16] U. M. Maurer and Y. Yacobi, "Non-interactive public-key cryptography," in *Advances in Cryptology (EUROCRYPT'91)*, pp. 498–507, Springer, 1991.

[17] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of Cryptology*, vol. 13, no. 3, pp. 361–396, 2000.

[18] S. S. D. Selvi, S. S. Vivek, V. K. Pradhan, and C. P. Rangan, "Efficient certificateless online/offline signature," *Journal of Internet Services and Information Security*, vol. 2, no. 3/4, pp. 77–92, 2012.

[19] S. S. D. Selvi, S. S. Vivek, V. K. Pradhan, and C. P. Rangan, "Efficient certificateless online/offline signature with tight security," *Journal of Internet Services and Information Security*, vol. 1, no. 1/2, pp. 115–137, 2013.

[20] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology*, pp. 47–53, Springer, 1984.

[21] A. Shamir and Y. Tauman, "Improved online/offline signature schemes," in *Advances in Cryptology (CRYPTO'01)*, pp. 355–367, Springer, 2001.

[22] H. Tanaka, "A realization scheme for the identity-based cryptosystem," in *Advances in Cryptology (CRYPTO'87)*, pp. 340–349, Springer, 2006.

[23] S. Tsujii and T. Itoh, "An id-based cryptosystem based on the discrete logarithm problem," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 4, pp. 467–473, 1989.

[24] Z. Wang and W. Chen, "An id-based online/offline signature scheme without random oracles for wireless sensor networks," *Personal and Ubiquitous Computing*, vol. 17, no. 5, pp. 837–841, 2013.

[25] T. S. Wu, Y. S. Chen, and K. Y. Lin, "Id-based online/offline signature from pairings," in *International Computer Symposium (ICS'10)*, pp. 198–203, 2010.

[26] S. Xu, Y. Mu, and W. Susilo, "Online/offline signatures and multisignatures for aodv and dsr routing security," in *Information Security and Privacy*, pp. 99–110, 2006.

[27] S. Xu, Y. Mu, and S. Willy, "Efficient authentication scheme for routing in mobile ad hoc networks," in *Embedded and Ubiquitous Computing (EUC'05)*, pp. 854–863, 2005.

[28] M. Yang and Y. Wang, "Improved identity based online/offline signature scheme," in *Ubiquitous Intelligence & Computing and 7th International Conference on Autonomic & Trusted Computing (UIC/ATC'10)*, pp. 126–131, 2010.

[29] W. S. Yap, S. H. Heng, and B. M. Goi, "An efficient certificateless signature scheme," in *Emerging Directions in Embedded and Ubiquitous Computing*, pp. 322–331, 2006.

[30] Z. Zhang, D. S. Wong, J. Xu, and D. Feng, "Certificateless public-key signature: security model and efficient construction," in *Applied Cryptography and Network Security*, pp. 293–308, 2006.

Dan Liu is a Master Candidate at the School of Computer Science and Technology, Anhui University, China. Her research interests include digital signature, security and privacy for mobile wireless networks privacy protection, etc..

Shun Zhang is currently an Associate Professor and Master Advisor at the School of Computer Science and Technology, Anhui University, China. His research interests include secure multi-party computation, big data, etc..

Hong Zhong is a Professor and PhD Advisor at the School of Computer Science and Technology, Anhui University, China. Her research interests include security protocols, wireless sensor networks, etc..

Runhua Shi received the PhD degree from University of Science and Technology of China in 2011. He is currently a Professor with Anhui University, and a visiting fellow at the School of Computer Science and Software Engineering, University of Wollongong. His current research interest includes classical and quantum cryptography, in particular, privacy-preserving multiparty computations.

Yimin Wang is a PhD Candidate at the School of Computer Science and Technology, Anhui University, China. His research interests include security and privacy for wireless networks, cloud computing, big data, etc..

Automatic Generation of Security Protocol Implementations Written in Java from Abstract Specifications Proved in the Computational Model

Bo Meng[1], Chin-Tser Huang[2], Yitong Yang[1], Leyuan Niu[1], and Dejun Wang[1]

(Corresponding author: Bo Meng)

School of Computer, South-Central University for Nationalities, Wuhan 430074, China[1]

182, Minyuan Road, Hongshan District, Wuhan, Hubei Province 430074, P.R.China

mengscuec@gmail.com

Department of Computer Science and Engineering, University of South Carolina, Columbia 29208, USA[2]

huangct@cse.sc.edu

(Email: mengscuec@gmail.com)

(Received Dec. 7, 2015; revised and accepted Feb. 18 & March 25, 2016)

Abstract

In order to deploy security protocols in practice, it is highly important that they be implemented in a secure manner. In this study, we first introduce a model for code generation from abstract security protocol specifications. Then, we present the development of an automatic generator called CV2JAVA, which is able to translate a security protocol abstract specified in Blanchet calculus in the computational model into an implementation written in JAVA. Finally, we also use the automatic generator CV2JAVA and CryptoVerif to generate the Secure Shell Version 2(SSHV2) security protocol implementation written in JAVA from the SSHV2 security protocol implementation written in Blanchet calculus proved in the computational model.

Keywords: Code generation, code security, protocol security, security protocol

1 Introduction

Over the last twenty years, researchers including developers and users have worked a great deal on the analysis and verification of security protocol abstract specifications [10, 17]. However, we know that the final objective is to have the security protocol implemented using programming languages, for example, the Java language, to put it into practice in the information system. Hence, we need to research the methods for generating the codes for security protocols because proof of the security properties of security protocol abstract specifications is not enough to instill confidence in developers and users with regard to the degree of security during execution. In addition, even if we prove the security properties of security protocol abstract specifications, their implementation procedures may be error prone and insecure. Therefore, it is essential to analyze and prove the security properties of security protocol implementations.

Generally, there are two major approaches to implement security protocols. One approach is suitable for the legacy codes of security protocols. This approach can be divided into two types. One type is mainly based on the technologies of program analysis and verification, for example, logic and type theory, which are directly used to automatically verify cryptographic security. In addition, it also depends on adding many annotations and predicates/assertions to the security protocol implementations. The other type is model extraction, which first extracts the security protocol abstract specification from security protocol implementations and then uses the security protocol analyzer to analyze or prove the cryptographic security of security protocol implementations. The drawback of this approach is that it is difficult to implement correctly.

The other approach is code generation, which is suitable for obtaining new implementations of some security protocols. If we have developed the security protocol abstract specification and verified its security properties, we can use the code generation method to obtain the secure security protocol implementations written in programming languages. In the code generation method, the first step is to produce the security protocol abstract specifications represented in formal language, for example, by using pi calculus or probabilistic process calculus. Then, a security protocol analyzer/prover, for example, ProVerif or CryptoVerif, is used to analyze the securities of the security protocol abstract specification. Finally, a transformer is developed to transform the security protocol abstract specifications into the security protocol implementation.

Two types of models, namely, the symbolic model and computational model, have been proposed and applied for code generation. There have been some existing approaches based on the symbolic model, including [16, 3, 11, 14, 8, 15, 2, 9, 1, 18, 13], which will be reviewed and discussed in section 2, the related work section. The symbolic model and the computational model are complementary ways for security analysis of security protocols. However, we know that one limitation of the symbolic model is that it cannot instill confidence in users with regard to its security because in the symbolic model, the cryptographic primitives are highly abstracted as black boxes, which often result in some attacks being missed. In contrast, the computational model is based on complexity and probability and uses a strong notion of security to counter all probabilistic polynomial-time attacks; therefore, its security properties are practically verifiable and it can find some attacks missed in the symbolic model. Some existing schemes [5, 6, 7] using the computational model are introduced in the related work section. However, to the best of our knowledge, none of the existing schemes can achieve mechanized generation of security protocol implementations in Java from security protocol abstract specifications proved in the computational model. Thus, in this study, we choose to use the computational model and focus on the mechanized generation of security protocol implementation in Java from security protocol abstract specifications proved in the computational model.

The main contributions of this study are summarized as follows:

1) We give a brief survey of the state-of-art schemes for automatic generation of security protocol implementations based on both the computational model and the symbolic model. We find that there is no scheme for automatic generation of security protocol implementations in Java from security protocol abstract specifications proved in the computational model.

2) We introduce a model of code generation in which the security protocol implementations in source language SP [S], for example, Blanchet calculus, are translated into the security protocol implementations in a target language SP[T], for example, the Java language. It is also necessary to prove that for the adversary Adv[S] that is constructed based on any adversary Adv[T], if the security protocol implementations in source language SP [S] are secure, for any adversary Adv[T], the security protocol implementations in target language SP[T] are also secure .

3) We develop an automatic generator CV2JAVA that can transform the security protocol abstract specifications, which are the inputs of CryptoVerif to the security protocol implementations written in the JAVA language. We use CryptoVerif to prove abstract security protocol implementations in Blanchet calculus in the computational model and then use the automatic generator CV2JAVA developed by us to generate the SSHV2 security protocol implementation written in JAVA.

4) We use the automatic generator CV2JAVA and CryptoVerif to generate the SSHV2 security protocol implementation written in JAVA. First, we develop the SSHV2 security protocol implementation in Blanchet calculus, and then CryptoVerif is used to conduct the analysis of authentication of a SSHV2 security protocol implementation written in Blanchet calculus; finally, we use the automatic generator CV2JAVA developed by us to generate the SSHV2 security protocol implementation written in JAVA.

2 Related Work

In this section, we give an overview of the state of the art of mechanized generation of security protocol implementations from abstract security protocol specifications proved in two major models: the symbolic model and computational model. According to related references, there is no scheme or tool for the mechanized generation of security protocol implementation in the Java language from abstract security protocol specifications proved in the computational model.

In the symbolic model, Pironti and Sisto [16] develop an automatic framework called Spi2Java, which can translate the abstract security protocol specifications written in typed Spi calculus to the security protocol implementations based on some special libraries written in Java. In addition, if the special libraries have met certain requirements, the generated Java implementation correctly simulates the typed Spi calculus specification. Following the work of Pironti and Sisto [16], Pironti et al. [14] first improve the Spi2Java framework and then develop a semi-automatic tool to generate the security protocol implementations; Copet et al. [8] present a visual user interface approach SPI2JAVAGUI based on the automatic framework Spi2Java to generate and verify the security protocol implementations; Pironti and Sisto [15] also present an approach based on the automatic tool FS2PV to analyze the security of the data transformation function in protocol code written in a function language. Avalle et al. [2] design a framework called JavaSPI in which the Java language is not only used as a modeling language but also as the implementation language in the security protocol. In essence, it translates the abstract security protocol specifications written in Java into the inputs, which are analyzed by ProVerif, a resolution-based theorem prover for security protocols. Based on the work of Avalle et al. [2], Copet and Sisto [9] develop an automated tool based on the JavaSPI framework to generate the Java protocol code. Avalle et al. [1] present a survey on the methods of generating the security protocol code from the view of protocol logic and note that there are two main approaches: model extraction and code generation.

Backes et al. [3] introduce a new code generator Expi2Java that can translate the abstract security protocol specifications written in an extensible variant of the Spi calculus and is analyzed by ProVerif into the security protocol implementations written in Java, which has the features of concurrency, synchronization between threads, exception handling and a type system. At the same time, they formalized the translation algorithm of Expi2Java with the Coq proof assistant and proved that if the original models are well-typed, the generated programs are also well-typed. Li et al. [11] propose a multi-objective-language-oriented automatic code generation scheme for security protocols based on XML that describes the formal model of the security protocol. Quaresma and Probst [18] present a protocol implementation generator framework based on the LySatool and a translator from the LySa calculus in the symbolic model into C or the Java language. Modesti [13] develops an AnBx compiler that accepts a special notation AnB to generate the protocol code. However, the work does not provide the proof of the soundness of the translation process.

As for the computational model, Cade and Blanchet [5, 6, 7] develop a compiler that takes an abstract specification of the protocol as the input language of the computational protocol verifier CryptoVerif and translates it into an OCaml implementation. They also prove that this compiler preserves the security properties proved by CryptoVerif, and if the abstract protocol specification is proved secure in the computational model by CryptoVerif. Li et al. [12] develop an automatic verifier SubJAVA2CV, which is able to transform security protocols written in SubJAVA to the security protocol abstract specification written in Blanchet calculus in the computational model. However, as we mentioned earlier, none of these schemes can provide mechanized generation of security protocol implementations in Java from security protocol abstract specifications proved in the computational model, which is what we aim to achieve in this study.

3 The Model of Code Generation from Abstract Security Protocol Specifications

For the code generation in Figure 1, we should prepare the security protocol implementations SP[S] written in formal languages, such as Pi calculus, Blanchet calculus, and SPI calculus; while the target language of the security protocol implementations SP[T] are programming languages, for example, Java, C #, and the C language. This means that the target languages are programming languages, and the source languages are formal languages. If the code generation method is used to generate security protocol implementations in target languages, there are two requirements that need to be satisfied:

1) The semantic of the target language simulates the semantic of the source language;

2) For the adversary Adv[S] constructed based on any adversary Adv[T], if the security protocol implementation written in source language SP[S] is secure, for any adversary Adv[S], the security protocol implementation written in target language SP[T] is also secure.

The first requirement shows that from the view of the behaviors, the relationship between the security protocol implementations written in target language SP[T] and the security protocol implementations written in source language SP[S] is simulation or observational equivalence.

The second requirement shows how to prove the security properties of the security protocol implementations written in target language SP[T]. For any adversaries in the context, we want to prove the cryptographic security properties of the security protocol implementations written in target language SP[T]. Thus, according to any adversary Adv[T] in the security protocol implementations written in target language SP[T], we construct an adversary Adv[S] in the security protocol implementations written in source language SP[S] and then prove that for the adversary Adv[S]. If the security protocol implementations written in source language SP[S] are secure, for any adversary Adv[T], the security protocol implementations written in target language SP[T] are secure.

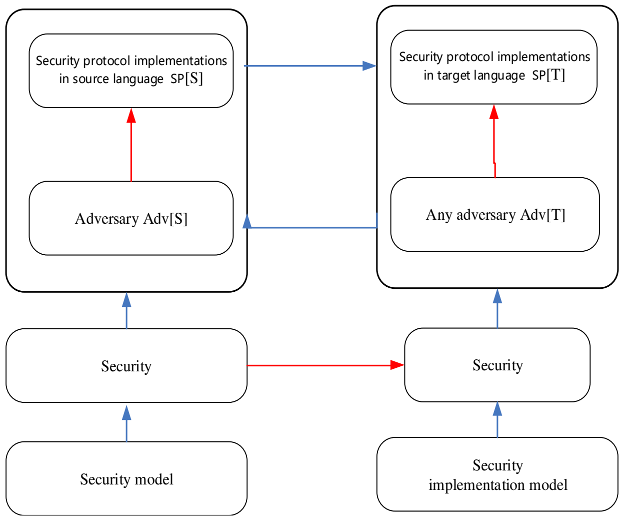

Security protocol implementations in source language SP[S]
Security protocol implementations in target language SP[T]
Adversary Adv[S]
Any adversary Adv[T]
Security
Security
Security model
Security implementation model

Figure 1: Model of code generation

4 Blanchet Calculus

Blanchet calculus [4] is based on probabilistic polynomial calculus and has been widely used to prove the security properties of security protocols. In Blanchet calculus, messages are modelled as bitstrings and cryptographic primitives are functions that operate on bitstrings. Blanchet calculus has two versions: channels front-end and oracles front-end in order to be as the input of CryptoVerif. Here we introduce the channels front-end version. Blanchet calculus is mainly made up of terms, conditions and processes. The adversary is modelled by an evaluation context Blanchet calculus mainly consists of terms and processes. Suppose there are m terms indexed as $1, \cdots, i, \cdots, m$; then, variable access $x[M_1, \cdots, M_i, \cdots, M_m]$ or function application $f(M_1, \cdots, M_i, \cdots, M_m)$ represent computations on bitstrings represented by these m terms. The variable access $x[M_1, \cdots, M_i, \cdots, M_m]$ outputs the content of the cell of indices $[M_1, \cdots, M_i, \cdots, M_m]$ of the m-dimensional array variable x. The function application $f(M_1, \cdots, M_i, \cdots, M_m)$ outputs the result of applying function f to $[M_1, \cdots, M_i, \cdots, M_m]$.

The processes in Blanchet calculus include the input process and output process. Input process 0 does nothing; $Q|Q'$ is the parallel composition of Q and Q'; $!^i$ represents n copies of Q in parallel; newChannel $c; Q$ produces a new private channel n and performs Q. Output process new $x[i_1, \cdots, i_m] : T; P$ chooses i_1, a new random number, uniformly, stores it in $x[i_1, \cdots, i_m]$, and then executes P. Random numbers are chosen by new $x[i_1, \cdots, i_m] : T$. Output process let new $x[i_1, \cdots, i_m] : T = M$ in P stores the bitstring value of M in $x[i_1, \cdots, i_m]$ and executes P.

Find process shows that it tries to find a branch J in $[1, m]$ such that there are values of $u_{j_1}, \cdots, u_{j_{m_j}}$ for which $M_{j_1}, \cdots, M_{j_{l_j}}$ are defined and M_j is true. In the case of success, it executes P. When event $e(M_1, \cdots, M_m)$ has been executed, the formula event $e(M_1, \cdots, M_m)$ is true.

5 Code Generation from Security Protocol Implementations Written in Blanchet Calculus

We know that the security protocol implementations written in Blanchet calculus are mainly composed of processes, and processes mainly consist of top processes and communicating party processes. The top process is the main process. The parties in security protocols are modelled as communicating party processes. Owning to the powerful ability of Blanchet calculus, the communicating parties in the security protocols are multiple parties. Here we classified it into two kinds of communicating parties: the sender processes and the receiver processes in the template for security protocol implementations written in Blanchet calculus as shown in Figure 2. In our model there can exist multiple sender parties and receiver parties. When we model the security protocol with Blanchet calculus, the problem needs to be addressed with different role names.

Generally the template file used to analyze the security protocols through Blanchet calculus is made up of the following sections: Type declaration, function declaration, cryptographic primitive declaration, channel declaration, security property, sender process, receiver process and top process. The type declaration section is composed of the related type in formalizing the security protocols, in which some types are owned by Blanchet calculus and other types are defined according to the requirement in formalizing the security protocols. The cryptographic primitives, for example, indistinguishability under adaptive chosen ciphertext attack public key enc, which are used in analyzing the security protocols, is declared in the Cryptographic primitive section. Channels are used to model the communication channel between the sender and receiver processes that are declared in the Channel declaration section. In the template, we assume that there are two roles in the security protocols: One is the sender, and the other is the receiver. Hence, the sender is modelled as the sender process in the Sender process section, and the receiver is modelled as the receiver process in the Receiver process section. The top process is composed of the sender process and the receiver process.

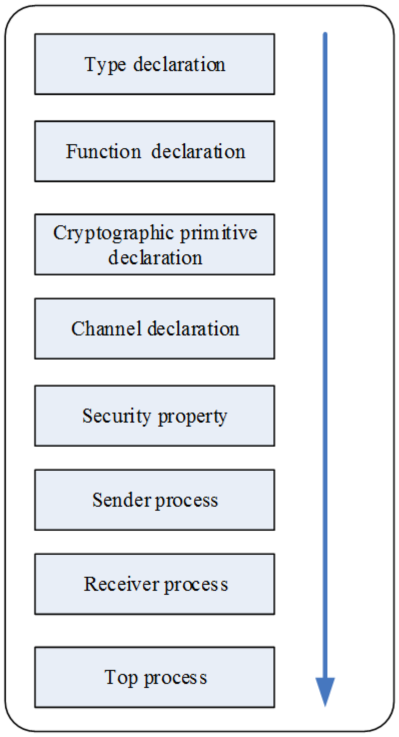


Figure 2: Template for security protocol implementations written in Blanchet calculus

Hence, in the translation of the code generator, the top process is mapped into the main program in security protocol implementations written in JAVA, which is a class. Regarding the type in Blanchet calculus, we directly transform it into the corresponding type in JAVA because in Blanchet calculus, we can define the required types for the types in JAVA showed in Table 1. The function in Blanchet calculus is translated into the function in JAVA. The cryptographic primitive in Blanchet calculus is transformed into the security package in JAVA. The communicating channel processes, which include input channels and output channels, are mapped into the classes, which are implemented based on the socket and are the parties in security protocol implementations written in JAVA. These classes are responsible for sending and receiving messages between the communicating parties. The sender process and the receiver process are mapped into the sender class and receiver class, respectively, in Figure 2. Figure 3 presents the generators from Blanchet calculus to JAVA in detail.

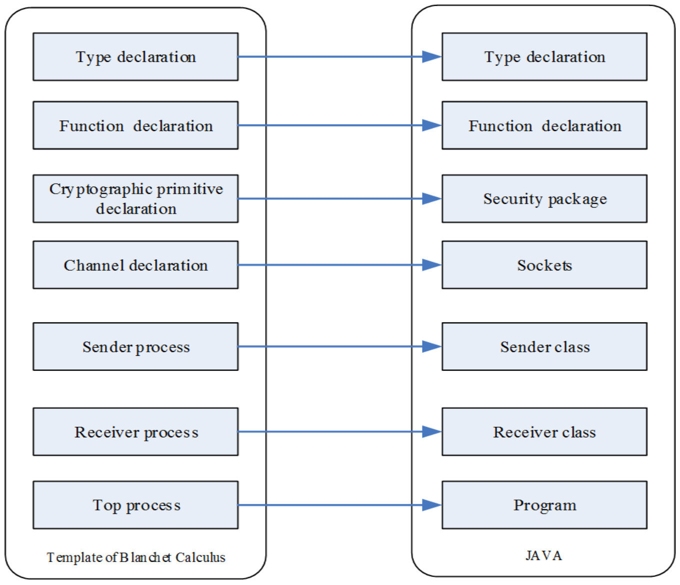


Figure 3: The model of code generation from Blanchet calculus

$$[\![x[M_1,\cdots,M_m]]\!] \triangleright \boxed{\begin{array}{l}\text{int[] x=new int[m];}\\ \text{x[M}_1\text{]; x[M}_2\text{]; x[M}_3\text{];...; x[M}_m\text{];}\end{array}}$$

Figure 4: GeneratorVariableAccess(x[:T])

Generation function GeneratorVariableAccess(x[:T]) in Figure 4 maps variable access x[:T] in Blanchet calculus into variable (int[] x= new int[m]; x[M$_1$]; x[M$_2$]; x[M$_3$];... x[M$_m$]) in JAVA.

Table 1: The transforming from some types in Blanchet calculus to some types in JAVA

Types in Blanchet calculus	Types in JAVA
keyseed	SecureRandom
key	Key
cleartext	String
ciphertext	String
seed	SecureRandom
mkeyseed	SecureRandom
mkey	key
macinput	String
macres	String
pkey	PublicKey
skey	PrivateKey
signinput	String
signature	byte[]
hashinput	String
hashoutput	String
input	int
output	int

$$[\![f[M_1,\cdots,M_m]]\!] \triangleright \boxed{\text{public Type FunName(Type1 M}_1\text{);}}$$

Figure 5: GeneratorFunctionApplication(f)

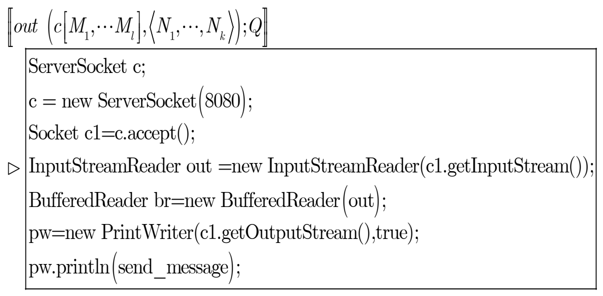
Generation function GeneratorFunctionApplication(f) in Figure 5 transforms function f in Blanchet calculus into function (public Type FunName(Type1 M$_1$);) defined by us in JAVA.

$$[\![in\ (c[M_1,\cdots,M_l],(x_1[\tilde{i}]:T_1,\cdots,x_k[\tilde{i}]:T_k));P]\!] \triangleright$$

```
ServerSocket c;
c = new ServerSocket(8080);
Socket c1=c.accept();
InputStreamReader in=new InputStreamReader(c1.getInputStream());
BufferedReader br=new BufferedReader(in);
pw=new PrintWriter(c1.getOutputStream(),true);
while(true) {
receive_message =br.readLine();
}
```

Figure 6: GeneratorInput in

Regarding the generation of input process in Blanchet calculus, we mainly use the network statements, for example, socket statement, in JAVA to implement it. A special socket is defined to implement the input process. The property of the input process socket of the sender is the address and port number of the sender. The property of the input process socket of the receiver is the address and port number of the receiver. Hence, input process in() in Blanchet calculus can be translated into JAVA as shown in Figure 6.

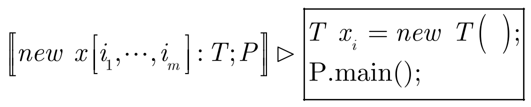


$[\![out\ (c[M_1,\cdots M_l],\langle N_1,\cdots,N_k\rangle);Q]\!] \triangleright$

```
ServerSocket c;
c = new ServerSocket(8080);
Socket c1=c.accept();
InputStreamReader out =new InputStreamReader(c1.getInputStream());
BufferedReader br=new BufferedReader(out);
pw=new PrintWriter(c1.getOutputStream(),true);
pw.println(send_message);
```

Figure 7: GeneratorOutput out

Regarding the generation of output process out in Blanchet calculus, we also use the network statements in JAVA to implement it. A special socket is defined to implement the output process. The property of the output process socket is the address and port number of the receiver. The property of the output process socket of the receiver is the address and port number of the sender. Hence, output process out in Blanchet calculus can be translated into JAVA showed in Figure 7.

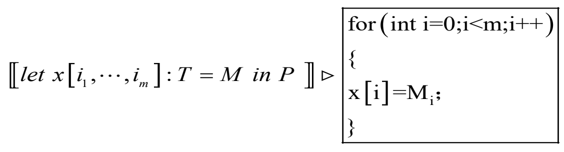


$[\![new\ x[i_1,\cdots,i_m]:T;P]\!] \triangleright$

```
T x_i = new T( );
P.main();
```

Figure 8: Generator new x

Regarding the process new x in Blanchet calculus, we mainly use the class new in JAVA to implement it. Hence, the new process in Blanchet calculus is transformed into JAVA code in Figure 8.

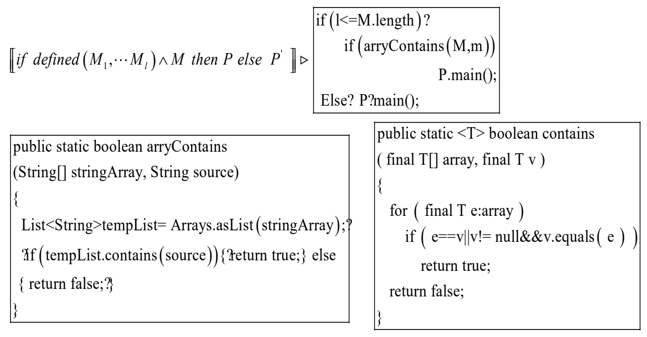


$[\![let\ x[i_1,\cdots,i_m]:T=M\ in\ P\ ]\!] \triangleright$

```
for(int i=0;i<m;i++)
{
x[i]=M_i;
}
```

Figure 9: GeneratorAssignment Let

The assignment process let x: T=M in P is transformed into the statementsfor (int i=0; i¡i++)x[i]=M$_i$in JAVA in Figure 9.



$[\![if\ defined(M_1,\cdots M_l)\wedge M\ then\ P\ else\ P']\!] \triangleright$

```
if (l<=M.length)?
    if(arryContains(M,m))
                    P.main();
Else? P?main();
```

```
public static boolean arryContains
(String[] stringArray, String source)
{
 List<String>tempList= Arrays.asList(stringArray);?
 ?if(tempList.contains(source)){?return true;} else
 { return false;?}
}
```

```
public static <T> boolean contains
( final T[] array, final T v )
{
  for ( final T e:array )
     if ( e==v||v!= null&&v.equals( e ) )
        return true;
  return false;
}
```

Figure 10: Generator additional if then

In the transformation of the conditional process if defined $(M_1,\cdots,M_1)M$ then P else P', there are two methods that can be used to implement the key part defined $(M_1,\cdots,M_1)M$. One method is that the array is first transformed into the list; then, we can use the method contained in the list to implement it. Hence, the arryContains is implemented in JAVA as:

```
public static boolean arryContains
        (String[] stringArray,String source)
{List¡String¿tempList=Array.asList(stringArray);
if (tempList.Contains(source)){return true;}
else {return false;}}
```

In the other method, the traversing array is used to implement it. So, the key part defined $(MM_1,\cdots,M_1)M$ is implemented in JAVA as in Figure 10:

```
public static boolean
        arryContains(final T[]array,fianl T v)
(final T e:array)if (e==v||v!=null&&v.equals(e))
return true;return false).
```

6 Automatic Generator CV2JAVA from Blanchet Calculus to JAVA

According to the generation functions introduced in the previous section, we develop an automatic generator CV2JAVA that accepts the security protocol model expressed by Blanchet calculus as an input and produces security protocol implementations in JAVA as an output.

Figure 11 presents the application of automatic generator CV2JAVA. First, we model the security protocols with Blanchet calculus, and then, the tool CryptoVerif is used to implement the analysis of security, and finally, the automatic generator CV2JAVA is used to generate the security protocol implementations written in JAVA.

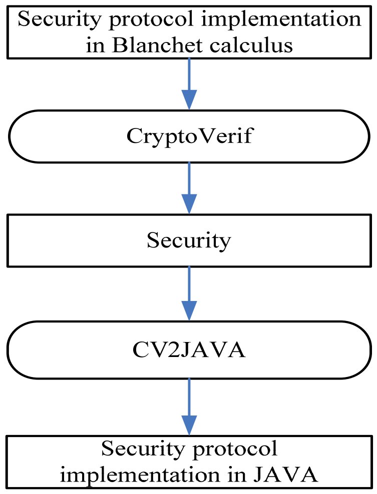

Security protocol implementation in Blanchet calculus
CryptoVerif
Security
CV2JAVA
Security protocol implementation in JAVA

Figure 11: Application of automatic generator CV2JAVA

Figure 12 shows the development of automatic generator CV2JAVA. First, according to the informal specification of the security protocol, we produce the security protocol implementations written in Blanchet calculus, and then, based on the syntax of Blanchet calculus, the lexical analyzer developed by us is used to analyze and verify the correctness of security protocol implementations written in Blanchet calculus. If verification succeeds, lexical elements, for example tokens, are produced. Next, the parser developed by us is used to process tokens and produces an abstract syntax tree to represent the structure of security protocol implementation written in Blanchet calculus. The goal is to generate secure security protocol implementations written in JAVA, so the structures and elements that are not related to the programming implementation of the security protocol, for example, irrelevant events, are deleted, and the simplified abstract syntax tree is produced. After this step, the translator mapping the simplified abstract syntax tree in Blanchet calculus into the abstract syntax tree in JAVA is derived.

Next, we develop the code generator to produce the security protocol implementations written in JAVA. In the next section, we mainly concentrate on the development of automatic generator CV2JAVA based on JavaCC. We use JavaCC to develop the lexical analyzer and parser. First, we need to construct the .jj file and then use JavaCC to implement the lexical analyzer and parser for Blanchet calculus. The .jj file is mainly composed of options, function declarations, specification for lexical analysis and BNF notations for Blanchet calculus.

6.1 Simplifier

Owing to the specialty of Abstract Syntax Tree of security protocol implementation written in Blanchet calculus, it is not easy to directly transform it into Abstract Syntax Tree of security protocol implementation written in JAVA. At the same time, it may produce some mistakes in the transformations.

To reduce Abstract Syntax Tree of security protocol implementation written in Blanchet calculus, we need to program a simplifier to perform the function. Simplifier accepts the Abstract Syntax Tree as inputs and produces the simplified Abstract Syntax Tree. Here, we implement the simplifier by using visitor pattern.

Visitor pattern is one type of design pattern. Visitor pattern defines an operation on the elements of an object structure. Without modifying the structure of the type, visitor pattern can access the different objects of a type and make different operations on them. In most cases, we do not add a new node in the Abstract Syntax Tree after it has been generated. Hence, we find that it is a good method to develop the simplifier visitor to address the Abstract Syntax Tree. At the same time, different operations can be made on the different nodes. Hence, it is proper to use visitor patterns to address this situation.

Simplifier visitor in Figure 13 visits, adds and deletes the nodes according to the Abstract Syntax Tree for JAVA. There are two different methods for processing the keywords in simplified visitors. One is the JAVA untransforming method. The other is the JAVA transforming method. The JAVA untransforming method visits the Abstract Syntax Tree for Blanchet calculus and obtains syntax keywords and then compares it to the syntax keywords in JAVA. If they are the same, the syntax keywords in Blanchet calculus are not modified. At the same time, it abstracts the main keyword nodes and variable nodes in Blanchet calculus to implement the simplification of the Abstract Syntax Tree. The JAVA transforming method visits the Abstract Syntax Tree for Blanchet calculus and obtains syntax keywords and then compares them to the syntax keywords in JAVA. If they are not same, the syntax keywords in Blanchet calculus are modified. At the same time, it abstracts the main keyword nodes and variable nodes in Blanchet calculus to implement the simplification of the Abstract Syntax Tree for the main key variable nodes.

JJTree has additional good support for the visitor design pattern. If we set the VISITOR option to true, a jjtAccept() method and childrenaccept() method are inserted into all of the node classes it generates by JJTree. Apart from that, a visitor interface Visitor.java is also produced that can be implemented and passed to the nodes to accept. Our simplifier is an instance of a visitor interface Visitor.java. JJTree generates SimpleNodes, and they are processed differently.

Here is the process of the statement new a:T;P in Blanchet calculus. It generates a new variable for which the type is T and then performs process P. There is also the key word new. Hence, we construct the simplifier visitor according to the new statement syntax in JAVA.

The syntax of the statement new a:T;P in Blanchet calculus is presented in Figure 14. The syntax of statement new in JAVA is shown in Figure 15.

According to the production of the new statement, the syntax tree of statement new a:T;P in Blanchet calculus

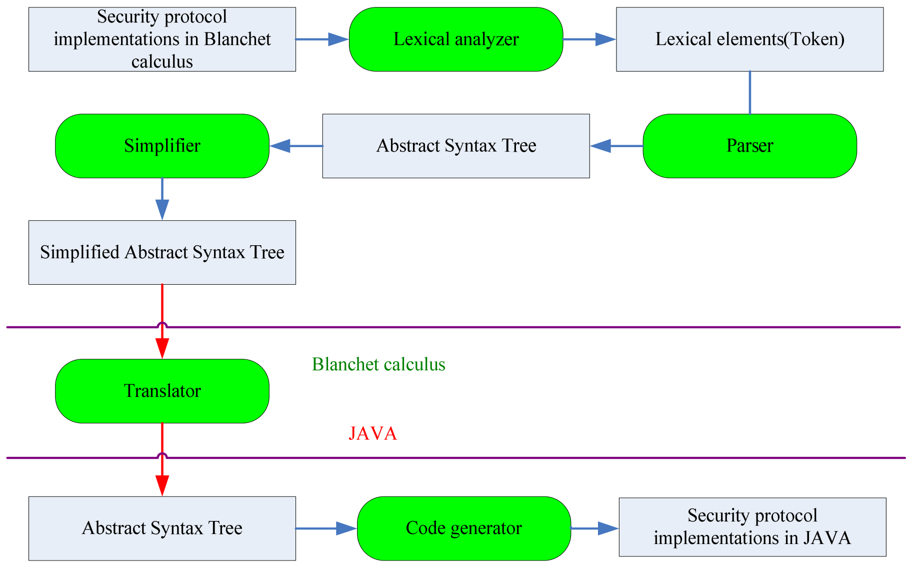


Figure 12: Development of automatic generator CV2JAVA

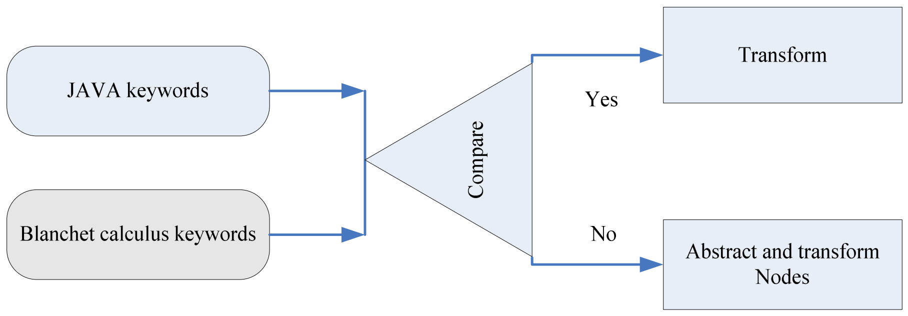


Figure 13: The idea of the simplifier visitor

```
void newterm():
{ }{
  new   vartype() scolon() term()
}
```

Figure 14: The syntax of statement new a: T; P in Blanchet calculus

is expressed in Figure 16. The key node is newex node, and the key variables are constant variable identex and type variable indentex.

Simplifier visitor travels the Syntax Tree of new statement in Blanchet calculus and abstracts the key nodes and adjusts the sequence of the key nodes of Syntax Tree for Blanchet calculus. Finally, it adds the correspondent Java nodes. The method is the JAVA untransforming method. The key word is new, and the node term() should be deleted by the case statement in Figure 17. The method used to delete the nodes is implemented in the method for TreeVisitor and can be used to travel the nodes of the syntax tree.

```
Type a = new Type();
```

Figure 15: The syntax of statement new in JAVA

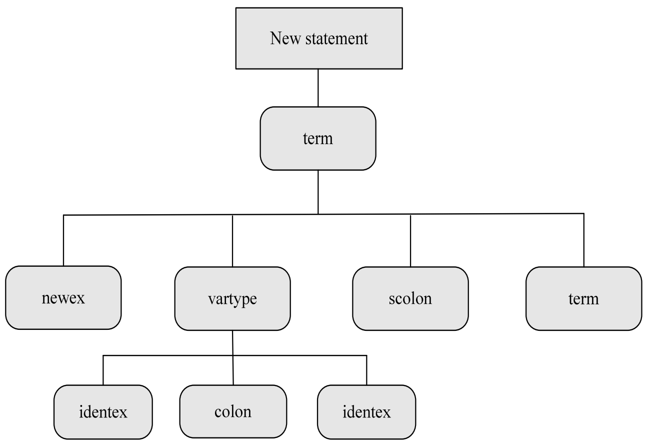


Figure 16: Syntax tree of new statement in Blanchet calculus

Based on the new statement syntax tree, TreeVisitor travels the nodes following the sequence. First, the jjtAccept(Visitor, Object) method of the new statement node is invoked by the main method in parser. Parameter Visitor is the object addvisitor. Then, the visit method of object addvisitor is called, and the new statement node is sent to visit method. The post-order traversal is applied. So, with parameter addvisitor, the visit method is called childrenAccept(Visitor, Object) in new statement node to travel to subnodes. The jjtAccept(Visitor, Object) method in subnodes to perform the traversal of the subtree is invoked by the childrenAccep method. Then, it invokes the addchildren() method to add new nodes. In addition, the children fields in the new node are revalued to avoid the logical error. At the same time, the fourth node is deleted by invoking the method visitor of the object deletevisitor.

```
case EG2TreeConstants.JJTNEWEX:
   node.childrenAccept(this, data);
   deletechildren(node, 4);
   break;
 case EG2TreeConstants.JJTNEWEX:
   node.childrenAccept(this, data);
   ((StringBuffer) data).append(node.jjtGetValue().toString());
   break;
```

Figure 17: Case statement

6.2 Translator

Translator is also a visitor that takes the simplified Abstract Syntax Tree for Blanchet calculus as an input and outputs the Abstract Syntax Tree for JAVA. This means that translator is a mapping function from language elements in Blanchet calculus to language elements in JAVA based on the definitions. In the next section, we show how to use the visitor to perform the mapping function GeneratorRandomNumber x, which implements mapping from a new statement in Blanchet to a new statement in JAVA.

We have used the simplifier to delete the node term(). To implement the translation from the new statement in Blanchet calculus and form the new statement in JAVA, we also need to adjust the position according to the syntax tree of the new statement in JAVA. The content node, identex node, colon node, sclon node and newex node are not changed. The identex node needs to be changed through the method exchangenode(node,1). According to the syntax nodes in JAVA, the method nodeExchange in translator can be used to exchange the nodes if the jjtNodeName of the node is the same as the predefined JJTLOCALVARIABLEDECLARATION. Finally, we generate the model for the Abstract Syntax Tree in Blanchet calculus shown in Figure 18.

When translator has visited all the nodes in the Abstract Syntax Tree for Blanchet calculus, the corresponding Abstract Syntax Tree for Blanchet calculus will be generated according to the model for the Abstract Syntax Tree in Blanchet calculus in Figure 18.

Figure 19 shows the Abstract Syntax Tree for the new statement in JAVA.

6.3 Code Generator

If we have derived the Abstract Syntax Tree of Blanchet calculus, the next task is to implement the code generator to generate code of Blanchet calculus, which is the security protocol implementation written in Blanchet calculus. Although the Abstract Syntax Tree mainly consists of the type name, variable name, method name and parameter list, we are not able to directly use the leaf node to generate security protocol implementation in JAVA because

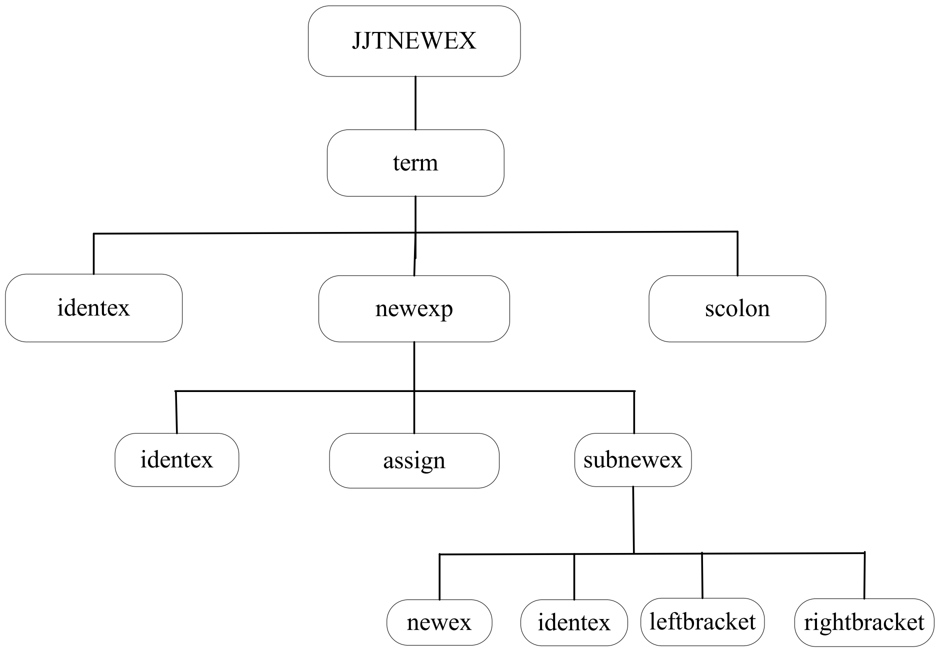


Figure 18: The model for the abstract syntax tree of code exchange method in Blanchet calculus

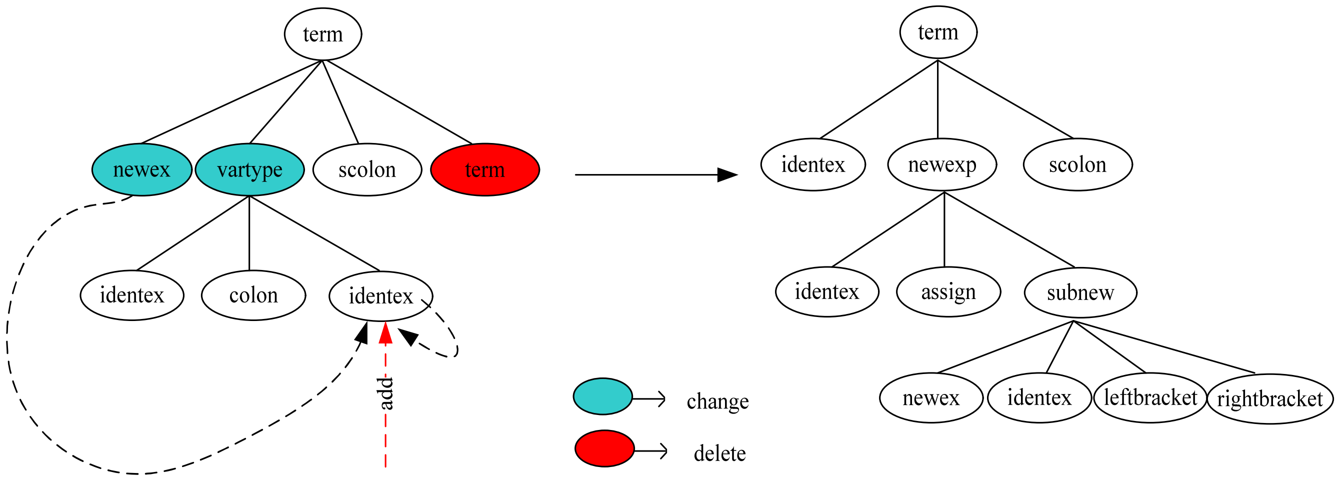


Figure 19: Abstract syntax tree of new statement

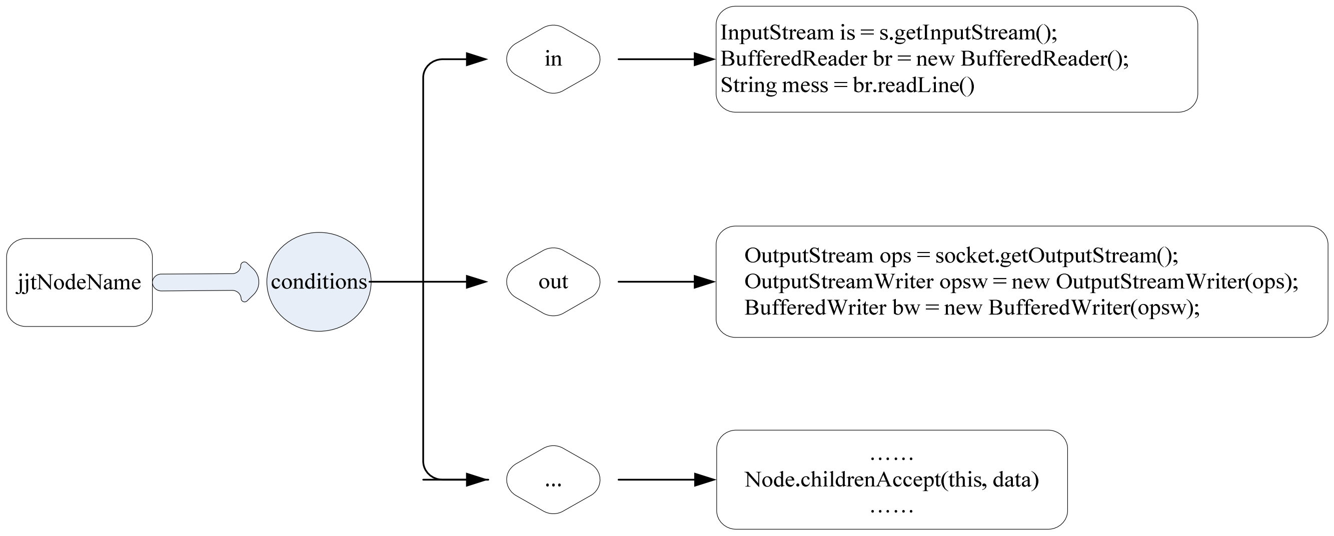

jjtNodeName
conditions
in
out
...
InputStream is = s.getInputStream();
BufferedReader br = new BufferedReader();
String mess = br.readLine()
OutputStream ops = socket.getOutputStream();
OutputStreamWriter opsw = new OutputStreamWriter(ops);
BufferedWriter bw = new BufferedWriter(opsw);
......
Node.childrenAccept(this, data)
......

Figure 20: The structure of the code generator visitor

the channel declaration, security definition and declaration have not been transformed. Hence, we design a visitor code generator for channel declaration, security definition and declaration. When the code generator traverses the syntax tree and adds input channels, out channels, operators and security definitions and declarations are in the proper position.

When the code generator traverses the syntax tree, it first adds the declaration methods and security properties verification and then translates it into JAVA code. For example, to verify the security of sessionkey, the statement in Blanchet calculus is query secret sessionkey; therefore, the corresponding method verifSecret(sessionkey) in JAVA is added to verify the sessionkey.

Figure 20 shows the structure of the visitor code generator. The visitor method first accesses the jjtNodeName of nodes and then creates different operations on different nodes. The switch statement is used to match the value of the jjtNodeName, and the task of processing the node is performed in the case statement. It mainly adds some keywords, operators and special characters and so on into the object StringBuffer.

Security protocol implementations in JAVA is stored in the object StringBuffer after the code generator visitor travels through all of the Abstract Syntax Tree. Finally, we write the contents in the object StringBuffer into a file, and thus, the security protocol implementation in JAVA is completed.

6.4 Templator

Templator is used to add the security object expressed by events in Blanchet calculus into security protocol implementation in CryptoVerif. Thus, we can use CryptoVerif to verify the security properties of security protocol implementation in Blanchet calculus.

7 Case: SSHV2 Security Protocol Implementation in JAVA

In this section, we use the automatic generator CV2JAVA and CryptoVerif to analyze the authentication of the Secure Shell Version 2 (SSHV2) security protocol and generate its implementation written in JAVA. We first develop the SSHV2 security protocol implementation written in Blanchet calculus. Then, we use the automatic verifier CryptoVerif to analyze the authentication of the SSHV2 security protocol. Finally, we use automatic generator CV2JAVA to generate SSHV2 security protocol implementation written in JAVA.

7.1 Review of the SSHV2 Security Protocol

The SSHV2 security protocol was issued in 2006 by IETF and is designed to implement remote secure login and other secure network services over an insecure network. The SSHV2 protocol is made up of three major sub-protocols: The Transport Layer Protocol implements server authentication, confidentiality, and integrity with perfect forward secrecy. The User Authentication Protocol authenticates the client identity to the server. The Connection Protocol provides the encrypted tunnel in several logical channels. The SSHV2 security protocol uses the Digital Signature Algorithm (DSA) to replace the RSA algorithm to implement the secret key exchange in the Transport Layer Protocol and use the Hash Message Authentication Code (HMAC) to guarantee the integrity of the message.

To implement the security of data over a public network, the message exchange is made up of the following four phases: the protocol version negotiation, encryption algorithms and key negotiation, key exchange and authentication. In the protocol version negotiation phase, the client and server chose the same version to communicate.

In the encryption algorithms and key exchange phases in the Transport Layer Protocol, parameter negotiation is allowed to minimize the number of round trips. The key exchange method, public key algorithm, symmetric encryption algorithm, message authentication algorithm, and hash algorithm are all negotiated. Hence, there are nine messages to be exchanged between the client and the server: Protocol Version Exchange between the client and the server SSH-protoversion-softwareversion, the client and server Algorithm Negotiation SSH_MSG_KEXINIT, key exchange request SSH_MSG_KEXDH_GEX_REQUEST, key exchange parameters response SSH_MSG_KEXDH_GEX_GROUP, key exchange parameter initialization SSH_MSG_KEXDH_GEX_INIT, key exchange response SSH_MSG_KEXDH_GEX_REPLY and the new key message SSH_MSG_NEWKEYS.

In the authentication phase, the server implements the authentication of the client. The server initiates the authentication by telling the client which authentication methods can be chosen to continue the exchange. The client can choose the proper methods listed by the server in any order. This gives the server complete control over the authentication process. There are three methods that can be used by the client: public key, password, and host-based client authentication methods. Here, we chose the public key authentication method.

In the public key authentication method, the client sends the Authentication Requests message to the server: (1) SSH_MSG_USERAUTH_REQUEST, which is made up of username, servicename, methodname and method_specific fields. The value of methodname is "public key" based on the authentication method we chose. The method_specific field consists of signature_algorithm, public key, signature fields. The signature field is made up of session ID,SSH_MSG_USERAUTH_REQUEST, user name, service, "public key", TRUE, algorithm name, public key fields.

When the server accepts authentication, it sends the message (2) SSH_MSG_USERAUTH_SUCCESS to the client; otherwise, it sends the messages SSH_MSG_USERAUTH_FAILURE SSH, which means that the authentication request has failed.

7.2 SSHV2 Security Protocol Implementation Written in Blanchet Calculus

The formal model implemented in Blanchet calculus includes initialization process, the client process and server process.

The initialization process generates the public key pkeyrsa and private key skeyrsa for the server, the signature public key signpkey, the signature private key signskey for the server, the public key pkey_c, private key skey_c for the client and the signature public key psignkey_c, the signature private key ssignkey_c for the client. Then, the public keys pkeyrsa, skeyrsa, signpkey, pkey_c, keyhash are published through the channel c. After that, the client process ClientProcess and the server process ServerProcess are launched.

The client process produces the version information clientversion and sends it through the channel c1. Then, it receives the version information messagetwo_s from the channel c4 and sets the version equal to messagetwo_s. The client process also generates the client algorithm parameters message_algoclient and sends it through the channel c5. Next, it receives the server algorithm parameters messagetwo_s from the channel c8 The client process accepts messagetwo_s. The client process generates the key exchange parameters minc, dp, maxc using the statements new minc: Z; new dp: vlue; new maxc: Z; it then sends them to the server process through channel c9. The random number r_c produced using the statement new r_c: Z accepts as the input of the function exp(); then, we can get the dh_e, which is sent to the server process through the channel c13. When the server receives the message dh_e, it will send a message dh_f_s as the response. Thus, through the channel c16, the client process gets the message dh_f_s generated by the server process and r_c to compute the client share key clientshareK using the function Gtokey(). The client process verifies the digital message dh_signature_s received from the server process using the function check(htomes(clienthash), psignkey_c, dh_signature_s) with the signature key psignkey_c.

If the verification is successful, the client authenticates the server. Next, the client process generates the new session key sessionkey using the function hashtokey(), and then, the new session key sessionkey is encrypted with the public keypkeyrsa of the server, and the ciphertext messenc is produced and is sent to the server process through the channel c17. The four parameters of the function hashtokey() are client share key clientshareK, the hash of client message clienthash, client message clientmes and session id sessionid. The session key sessionkey is the hash value of the four parameters: client shares key clientshareK, the hash of client message clienthash, client message clientmes and session id seesionid. Next, the client generates the authentication request message SSH_MSG_USERAUTH_REQUEST, which includes username, servicename, methodname, method_specific. Then, it uses the client signature key ssignkey_c to generate the digital signature request_signature of the authentication request message SSH_MSG_USERAUTH_REQUEST and send it to the server process through channel c19.

The server process receives the version information messageversion_c through channel c6. Then, it generates the server version information serverversion and sends it through channel c3. The server process also receives the client algorithm parameters message_algoc through the channel c2. Then, it generates the server algorithm parameters message_algoserver and sends it through the channel c7. The server receives the message message_DHrequ through the channel c10. The message message_DHrequ is made up of the key parameters dhmic, dhdp and dhmaxc, which are accepted as the inputs of function exp() to generate the message message_DHgroup and send it to the client process.

The server receives the message message_c and computes the shared key k through k=exp(message_e,r_s). Next, it calculates the hash value hashserver by the hash function hash(keyhash,messagedhhash). Then, the signature signtext is generated by fhashtosign(hashserver) and the client uses the signature private key ssingkey_c to produce the digital signature dh_signature. The server process generates the authentication message dhgexreply through the function concatreply, which accepts dh_f, pkeyrsa, session_id_s,dh_signature as the inputs. Finally, the server process sends the authentication message dhgexreply to the client through the channel c15.

The server process receives the authentication request message messageauth_c from the client process through the channel c20. Then, it constructs the message messageauth_c through the function concatsshuser(). Next, the server process uses the function check to verify the authentication request message with the public key cpkey_c. If the verification is successful, the server process sends the message SSH_MSG_USERAUTH_SUCESS through the channel c21.

7.3 Authentication of SSHV2 Security Protocol Implementation in Blanchet Calculus

In this section, we give a brief overview of the mechanized prover CryptoVerif, which is used to automatically analyze authentication of SSHV2 security protocol implementation in Blanchet calculus.

Here, we use non-injective correspondences in Figure 21 to model the authentication from server to client and from client to server. Event client(x, y, kx, px, gx, gy, krsa)==> server(x, y, kx, px, gx, gy, krsa) is used to authenticate the client by server. Event server A(ya)==>client A(ya) is used to authenticate the server by client. These events and non-injective correspondences had been added into the model implemented in Blanchet calculus by hand.

The analysis was performed by CryptoVerif and succeeded. The result is shown in Figure 22, and the SSHV2 security protocol is proved to guarantee authentication. "All queries proved" in Figure 22 shows that event client(x, y, kx, px, gx, gy, krsa)==> server(x, y, kx, px, gx, gy, krsa) is true, which shows that the server authenticates the client. Event server A(ya)==> client A(ya) is true, which shows that the client authenticates the server; query secret sessionkey is proved, which shows that the session key is secure.

```
event client(version,version,key,value,G,G,rsapkey).
event server(version,version,key,value,G,G,rsapkey).
event clientA(signature).
event serverA(signature).

query x:version,y:version,px:value,gx:G,gy:G,kx:key,krsa:rsapkey;
event client(x,y,kx,px,gx,gy,krsa)==>server(x,y,kx,px,gx,gy,krsa).
query ya:signature;
event serverA(ya)==>clientA(ya).
query secret sessionkey.
```

Figure 21: Events

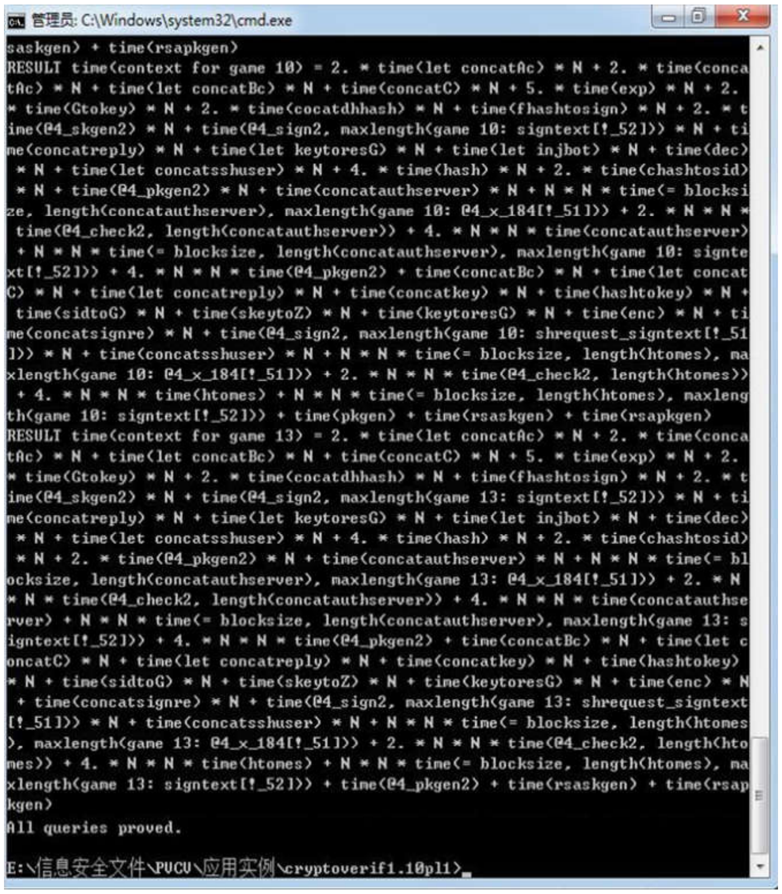
7.4 SSHV2 Security Protocol Implementation Written in JAVA

According to the SSHV2 protocol, a protocol implementation written in Blanchet Calculus, we use the automatic verifier CV2JAVA to generate the SSHV2 protocol implementation in JAVA, as shown in Figure 23. But in our current version, there are some limitations on it. we also need to implement its method in the class. Then, we run the client and server code, which is shown in Figure 24. In addition, it also shows that the server authenticates the client.

8 Conclusions

In the last twenty years, many security protocols have been introduced and deployed in all kinds of information systems. Hence, the verification of the security properties of these protocols has received plenty great deal of attention. Now, there is a popular issue: analysis of the security protocol implementations, which is introduced from the security field. In this study, we first present the model of code generation from abstract security protocol specifications. Then, we develop an automatic generator CV2JAVA, which is able to translate security protocol abstract specifications written in Blanchet calculus in the computational model into security protocol implementations written in JAVA. Moreover, we also use the automatic generator CV2JAVA and CryptoVerif to generate the SSHV2 security protocol implementation written in JAVA from the SSHV2 security protocol implementation written in Blanchet calculus proved in the computational model.

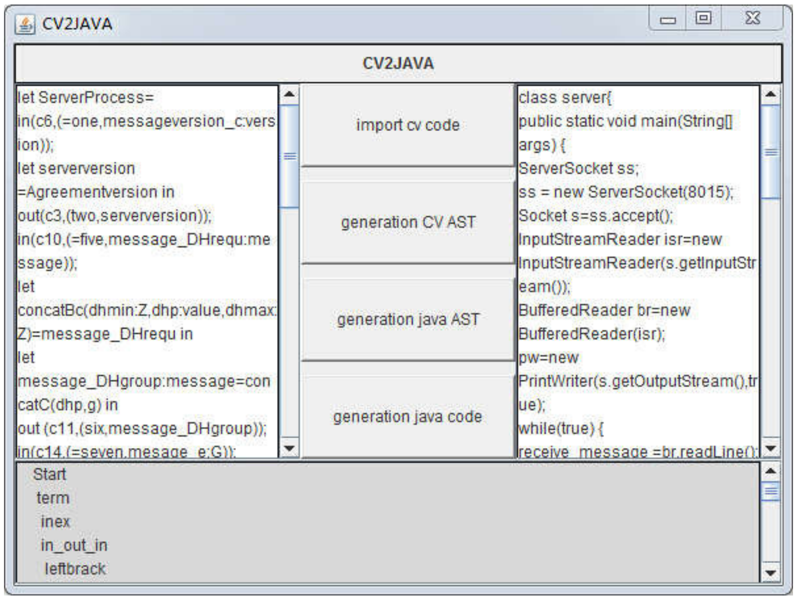
One of our main work is to develop a generator form security protocols implementations written in Blanchet calculus to JAVA code and is not to develop a complex complier. Hence the compile time error and compiler optimization etc. have not been addressed in current work. Compared to the works of Cade and Blanchet [12,13], there are six big difference. Firstly, the target languages



```
saskgen) + time(rsapkgen)
RESULT time(context for game 10) = 2. * time(let concatAc) * N + 2. * time(conca
tAc) * N + time(let concatBc) * N + time(concatC) * N + 5. * time(exp) * N + 2.
* time(Gtokey) * N + 2. * time(cocatdhhash) * N + time(fhashtosign) * N + 2. * t
ime(@4_skgen2) * N + time(@4_sign2, maxlength(game 10: signtext[!_52])) * N + ti
me(concatreply) * N + time(let keytoresG) * N + time(let injbot) * N + time(dec)
 * N + time(let concatsshuser) * N + 4. * time(hash) * N + 2. * time(chashtosid)
 * N + time(@4_pkgen2) * N + time(concatauthserver) * N + N * N * time(= blocksi
ze, length(concatauthserver), maxlength(game 10: @4_x_184[!_51])) + 2. * N * N *
 time(@4_check2, length(concatauthserver)) + 4. * N * N * time(concatauthserver)
 + N * N * time(= blocksize, length(concatauthserver), maxlength(game 10: signte
xt[!_52])) + 4. * N * N * time(@4_pkgen2) + time(concatBc) * N + time(let concat
C) * N + time(let concatreply) * N + time(concatkey) * N + time(hashtokey) * N +
 time(sidtoG) * N + time(skeytoZ) * N + time(keytoresG) * N + time(enc) * N + ti
me(concatsignre) * N + time(@4_sign2, maxlength(game 10: shrequest_signtext[!_51
])) * N + time(concatsshuser) * N + N * N * time(= blocksize, length(htomes), ma
xlength(game 10: @4_x_184[!_51])) + 2. * N * N * time(@4_check2, length(htomes))
 + 4. * N * N * time(htomes) + N * N * time(= blocksize, length(htomes), maxleng
th(game 10: signtext[!_52])) + time(pkgen) + time(rsaskgen) + time(rsapkgen)
RESULT time(context for game 13) = 2. * time(let concatAc) * N + 2. * time(conca
tAc) * N + time(let concatBc) * N + time(concatC) * N + 5. * time(exp) * N + 2.
* time(Gtokey) * N + 2. * time(cocatdhhash) * N + time(fhashtosign) * N + 2. * t
ime(@4_skgen2) * N + time(@4_sign2, maxlength(game 13: signtext[!_52])) * N + ti
me(concatreply) * N + time(let keytoresG) * N + time(let injbot) * N + time(dec)
 * N + time(let concatsshuser) * N + 4. * time(hash) * N + 2. * time(chashtosid)
 * N + 2. * time(@4_pkgen2) * N + time(concatauthserver) * N + N * N * time(= bl
ocksize, length(concatauthserver), maxlength(game 13: @4_x_184[!_51])) + 2. * N
* N * time(@4_check2, length(concatauthserver)) + 4. * N * N * time(concatauthse
rver) + N * N * time(= blocksize, length(concatauthserver), maxlength(game 13: s
igntext[!_52])) + 4. * N * N * time(@4_pkgen2) + time(concatBc) * N + time(let c
oncatC) * N + time(let concatreply) * N + time(concatkey) * N + time(hashtokey)
* N + time(sidtoG) * N + time(skeytoZ) * N + time(keytoresG) * N + time(enc) * N
 + time(concatsignre) * N + time(@4_sign2, maxlength(game 13: shrequest_signtext
[!_51])) * N + time(concatsshuser) * N + N * N * time(= blocksize, length(htomes
), maxlength(game 13: @4_x_184[!_51])) + 2. * N * N * time(@4_check2, length(hto
mes)) + 4. * N * N * time(htomes) + N * N * time(= blocksize, length(htomes), ma
xlength(game 13: signtext[!_52])) + time(@4_pkgen2) + time(rsaskgen) + time(rsap
kgen)
All queries proved.

E:\信息安全文件\PVCV\应用实例\cryptoverif1.10pl1>_
```

Figure 22: The result

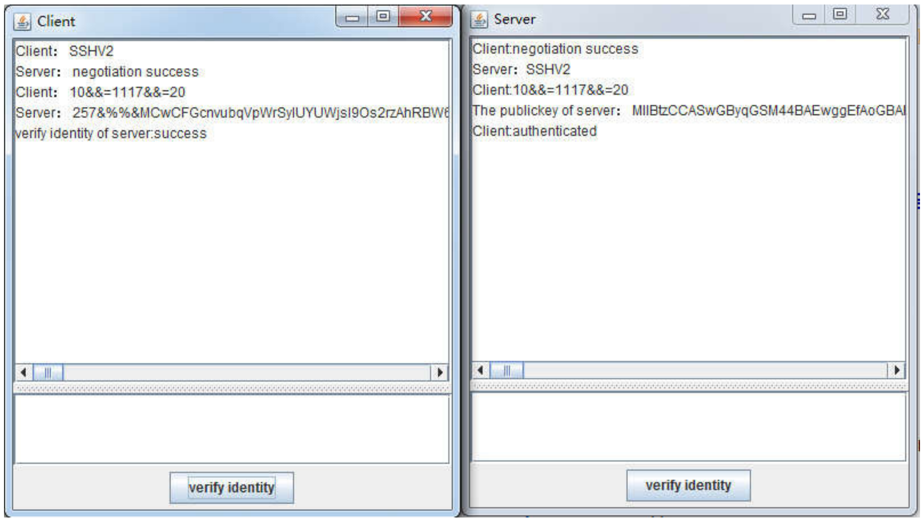


```
let ServerProcess=
in(c6,(=one,messageversion_c:vers
ion));
let serverversion
=Agreementversion in
out(c3,(two,serverversion));
in(c10,(=five,message_DHrequ:me
ssage));
let
concatBc(dhmin:Z,dhp:value,dhmax:
Z)=message_DHrequ in
let
message_DHgroup:message=con
catC(dhp,g) in
out (c11,(six,message_DHgroup));
in(c14,(=seven,mesage_e:G));
```

```
class server{
public static void main(String[]
args) {
ServerSocket ss;
ss = new ServerSocket(8015);
Socket s=ss.accept();
InputStreamReader isr=new
InputStreamReader(s.getInputStr
eam());
BufferedReader br=new
BufferedReader(isr);
pw=new
PrintWriter(s.getOutputStream(),tr
ue);
while(true) {
receive_message =br.readLine();
```

```
 Start
  term
   inex
   in_out_in
    leftbrack
```

Figure 23: Generating SSHV2 protocol implementation written in JAVA



```
Client:   SSHV2
Server:   negotiation success
Client:   10&&=1117&&=20
Server:   257&%%&MCwCFGcnvubqVpWrSylUYUWjsl9Os2rzAhRBW6
verify identity of server:success
```

```
Client:negotiation success
Server:  SSHV2
Client:10&&=1117&&=20
The publickey of server:   MIIBtzCCASwGByqGSM44BAEwggEfAoGBAI
Client:authenticated
```

Figure 24: The result of executing the client and server code

are different. Firstly, the target languages are different. The target language in Cade and Blanchet [12,13] is the Ocaml language which is a function language. By contrast, the target language in our work is Java language which is an imperative language. Secondly, the formal languages are different. In Cade and Blanchet [12,13], the modified Blanchet calculus is used, in which some statements are added and some other statements are removed in order to deal with transforming from it to Ocaml language. By contrast, we use the original Blanchet calculus to deal with transforming from it to JAVA language. Thirdly, the Blanchet calculus has two versions: channels front-end and oracles front-end as the input language of CryptoVerif. Cade and Blanchet [12] use the oracles front-end to model the SSH protocol. However, we use the channels front-end to model the SSH protocol. Hence the formal code are very different. Fourthly, the technologies based upon during the development of the translator may be different because of the different target languages. They use the Ocaml language to develop the complier. But we chose the JAVACC as the base in order to process the transformation from Blanchet calculus to JAVA language.. Fifthly, The SSH protocol modelled in our work is nice distinction. For example, the cryptographic primitives. Finally, the proof of the correctness of translator are different. Owning to the differences of the formal languages and the target languages, although the frameworks are similar, the technologies used are different. We will prove it from the view of operational semantics. Moreover, the operational semantics are different due to the different target.

In our current version of CV2JAVA, there are some limitations on it. At a time the only one process in Blanchet calculus can be translated into classes in JAVA code. At the same time we also need to implement its method in the class. In future work, we will improve the generator CA2JAVA and use the formal method to prove the correctness of translation from Blanchet calculus to JAVA and enhance the ability of the tool.

Acknowledgments

This work was supported by natural science foundation of Hubei Province under the Grants No. 2014CFB249 and by the foundation of China Scholarship Council.Bo Meng and Chin-Tser Huang contributed equally to this work.The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] M. Avalle, A. Pironti, and R. Sisto, "Formal verification of security protocol implementations: a survey," *Formal Aspects of Computing*, vol. 26, no. 1, pp. 99–123, 2014.

[2] M. Avalle, A. Pironti, R. Sisto, and D. Pozza, "The Java SPI framework for security protocol implementation," in *the Sixth International Conference on Availability, Reliability and Security*, pp. 746–751, Vienna,Austria, Aug. 2011.

[3] M. Backes, A. Busenius1, and C. Hritcu, "On the development and formalization of an extensible code generator for real life security protocols," in *Proceeding of The 4th International Conference on NASA Formal Methods*, pp. 371–387, Norfolk, USA, Apr. 2012.

[4] B. Blanchet, "A computationally sound mechanized prover for security protocols," *IEEE Transactions on Dependable and Secure Computing*, vol. 5, no. 4, pp. 193–207, 2008.

[5] D. Cade and B. Blanchet, "From computationally-proved protocol specifications to implementations," in *Proceeding of The Seventh International Conference on Availability, Reliability and Security*, pp. 204–208, Prague, Czech, Aug. 2012.

[6] D. Cade and B. Blanchet, "From computationally-proved protocol specifications to implementations and application to ssh," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 4, no. 1, pp. 4–31, 2013.

[7] D. Cade and B. Blanchet, "Proved generation of implementations from computationally secure protocol specifications1," *Journal of Computer Security*, vol. 23, no. 3, pp. 331–402, 2015.

[8] P. B. Copet, R. S. A. Pironti, D. Pozza, and P. Vivoli, "Visual model-driven design, verification and implementation of security protocols," in *The 14th IEEE International Symposium on High-Assurance Systems Engineering*, pp. 62–65, Omaha, Nebraska, USA., Oct. 2012.

[9] P. B. Copet and R. Sisto, "Automated formal verification of application-specific security properties," in *The 6th International Symposium on Engineering Secure Software and Systems*, LNCS 8364, pp. 45–59, Springer, 2014.

[10] A. Kartit, H. K. Idrissi, and M. Belkhouraf, "Improved methods and principles for designing and analyzing security protocols," *International Journal of Network Security*, vol. 18, no. 3, pp. 523–528, 2016.

[11] X. H. Li, D. Li, S. T. Li, and J. F. Ma, "Multi-language oriented automatic realization method for cryptographic protocols," *Journal on Communications*, vol. 33, no. 9, pp. 152–159, 2012.

[12] B. Meng, Z. M. Li, and W. Chen, "Mechanized verification of cryptographic security of cryptographic security protocol implementation in java through model extraction in the computational model," *Journal of Software Engineering*, vol. 9, no. 1, pp. 1–32, 2015.

[13] P. Modesti, "Efficient java code generation of security protocols specified in anb/anbx," in *the 10th International Workshop on Security and Trust Management*, pp. 204–208, Wroclaw, Poland, Sep. 2014.

[14] A. Pironti, D. Pozza, and R. Sisto, "Formally based semi-automatic implementation of an open security protocol," *Journal of Systems and Software*, vol. 85, no. 4, pp. 835–849, 2012.

[15] A. Pironti and R. Sisto, "Safe abstractions of data encodings in formal security protocol models," *Formal Aspects of Computing*, vol. 26, no. 1, pp. 125–167, 2014.

[16] A. Pirontia and R. Sisto, "Provably correct java implementations of spi calculus security protocols specifications," *Computers & Security*, vol. 29, no. 3, pp. 302–314, 2010.

[17] Q. Qian and Y. Long, J. R. Zhang, "A lightweight rfid security protocol based on elliptic curve cryptography," *International Journal of Network Security*, vol. 18, no. 2, pp. 354–361, 2016.

[18] J. Quaresma and C. Probst, "Protocol implementation generator," in *The 15th Nordic Conference on Secure IT Systems*, pp. 256–268, Espoo, Finland, Oct. 2010.

Bo Meng was born in 1974 in China. He received his M.S. degree in computer science and technology in 2000 and his Ph.D. degree in traffic information engineering and control from Wuhan University of Technology at Wuhan, China in 2003. From 2004 to 2006, he worked at Wuhan University as a postdoctoral researcher in information security. From 2014 to 2015, he worked at University of South Carolina as a Visiting Scholar. Currently, he is a full Professor at the school of computer, South-Center University for Nationalities, China. He has authored/coauthored over 50 papers in International/National journals and conferences. In addition, he has also published a book "secure remote voting protocol" in the science press in China. His current research interests include security protocols and formal methods.

Chin-Tser Huang received his Ph.D. in Computer Science at the University of Texas at Austin, Austin, Texas and is now an associate professor in the Department of Computer Science and Engineering at the University of South Carolina. He is also the director of the Secure Protocol Implementation and Development (SPID) Laboratory at the University of South Carolina. His current research interests include network security, network protocol design and verification, secure computing, and distributed systems.

Yitong Yang was born in 1991 and is now a postgraduate at the school of computer, South-Center University for Nationalities, China. Her current research interests include security protocols and formal methods.

Leyuan Niu was born in 1990 and currently is a postgraduate in the school of computer, South-Center University for Nationalities, China. Her current research interests include security protocols and formal methods.

Dejun Wang was born in 1974 and received his Ph.D. in information security at Wuhan University in China. Currently, he is an associate professor in the school of computer, South-Center University for Nationalities, China. He has authored/coauthored over 20 papers in international/national journals and conferences. His current research interests include security protocols and formal methods.

Minimizing Turtle-Shell Matrix Based Stego Image Distortion Using Particle Swarm Optimization

Qiang Jin[1], Zhihong Li[1], Chin-Chen Chang[2], Anhong Wang[1], and Li Liu[1]

(Corresponding author: Chin-Chen Chang)

Institute of Digital Multimedia and Communication[1]

Taiyuan University of Science and Technology, Taiyuan 030024, China

Department of Information Engineering and Computer Science[2]

Feng Chia University, Taichung 40724, Taiwan

No. 100, Wenhwa Rd., Seatwen, Taichung 40724, Taiwan

(Email: alan3c@gmail.com)

(Received Nov. 19, 2015; revised and accepted Feb. 16 & Mar 6, 2016)

Abstract

The purpose of data hiding is to embed secret messages into cover media so that the receiver will not be aware of the existence of these important messages. Recently, Chang et al. proposed a novel data hiding scheme based on turtle shells, which provided a good visual quality and an acceptable embedding capacity. However, the matrix that is used in the scheme based on turtle shells is not an optimal matrix. This means that the distortion of the image can be reduced further. In this paper, we propose a method that uses particle swarm optimization (PSO) to further reduce the distortion of the cover image based on turtle shells. Our experimental results confirmed that our scheme was efficient in finding an optimal replacement table and achieved a better visual quality of the cover image. Also, our scheme enhanced the security of scheme based on the turtle-shell matrix.

Keywords: Data hiding, GA, PSO, turtle shell

1 Introduction

With the rapid development and extensive application of the Internet, information security has been recognized as an important issue by many users. As a result, data encryption and data hiding, the two key techniques in the field of information security, also have received greater interest as topics of concern and research. Data encryption [3, 13] can alter a piece of clear text, or unencrypted information, into cipher text, or encrypted information. However, assailants are attracted easily by multimedia information after encrypting, and its content is totally transparent once the cipher-text is decrypted. Different from data encryption, data hiding is a technique that hides secret information in an image to make the secret data inaudible or invisible for assailants. Due to this advantage, it is used extensively in many fields, such as electronic commerce, copyright protection, and image authentication.

Generally, date hiding techniques are conducted mainly in three domains, i.e., the compression domain, the frequency domain and the spatial domain. In the compression domain, vector quantization (VQ) [5, 8, 9, 12] is often used in data hiding schemes to obtain more space for embedding secret information. In 2003, Du and Hsu [9] proposed an adaptive algorithm to embed secret data into VQ compressed images. Later, Hu [12] and Chang et al. [8] improved the embedding capacity of VQ compressed images to embed secret data. Although these compression-domain data hiding schemes achieved higher hiding capability, the reduction in the quality of the images was a serious concern.

In the frequency domain, first, cover images are transformed by using a frequency-oriented mechanism such as the discrete cosine transformation (DCT) [6, 16], or the discrete wavelet transformation (DWT) [1, 17]. Then the secret data are embedded into the transformed coefficients. Although these data hiding schemes can obtain the stego images that have better visual quality, the embedding capacity of secret data still is not satisfactory because only a small number of coefficients are used to embed the secret data.

In the spatial domain, secret messages are embedded directly into the pixels to ensure a high embedding capacity of secret data. The most common method is least significant bit (LSB) replacement. In the traditional LSB method [2], the LSB of the cover image is replaced directly by a secret bit. In 2001, Wang [22] proposed a data hiding method by using optimal LSB substitution and genetic algorithm to improve the visual quality of stego images. To further improve the hiding capacity and visual quality of stego images, Mielikainen [20] proposed an LSB matching scheme that uses a pair of pixels as a unit to embed two secret bits. However, the direction of modification on the cover pixels is exploited incompletely for data hiding in this scheme. In 2006, Zhang and Wang [23] proposed a data hiding scheme that fully exploits the modification directions (EMD) to overcome this problem. In 2008, hang et al. [4] proposed Sudoku solutions to guide the modification of pixels for data hiding to make the data more secure than [23] and [4]. Later, in order to improve the visual quality of Chang et al.'s method, Hong et al. [10] proposed the Sudoku-S scheme based on a search algorithm. In 2010, Kim et al. [15] proposed two new schemes, named EMD-2 and 2-EMD to enhance the embedding capacity of the original EMD scheme. However, these schemes offered an unacceptable trade-off between low image quality and restricted embedding capacity.

Recently, Chang et al. [7] proposed a novel data hiding scheme to overcome this problem by using turtle shells for guiding the modification of cover pixels. This scheme uses a hexagon, which is called a turtle shell, to contain eight different digits, ranging from 0 to 7. A matrix composed of a large number of turtle shells is used to alter the values of pairs of cover pixels for the purpose of hiding secret bits. This scheme achieves a hiding capacity of 1.5 bits per pixel and a peak signal-to-noise ratio (PSNR) value of about 49 dB. Later, Liu et al. [18] enhanced the embedding capacity by using the location table to develop the turtle shell-based scheme. However, the turtle-shell matrix in their schemes is not optimal, meaning that the distortion of the image can be reduced further. There are some well-known evolutionary algorithms that can be used to solve the optimization problem, such as genetic algorithm (GA) [11] and particle swarm optimization (PSO) [19]. Compared with GA, the main advantage of PSO is its high computational efficiency. In this paper, we propose a method that involves using PSO to minimize the distortion of the cover image based on turtle shells. In our method, first, an original matrix is generated according to Chang et al.'s scheme [7]. Then PSO is used to search the optimal replaceable table of the eight integers in the matrix. After a near optimal table is developed, a new matrix is generated by replacing all of the eight integers with integers from the near optimal table. Finally, the data hiding procedure is conducted according to the new matrix in the same way as was done in Chang et al.'s scheme. The experimental results confirmed that our scheme achieved smaller image distortion than the scheme in [7].

The rest of this paper is organized as follows. The two important techniques are reviewed in Section 2, and the proposed scheme is presented in detail in Section 3. The experimental results are discussed in Section 4, and our conclusions are presented in Section 5.

2 Related Work

2.1 A Novel Scheme for Data Hiding Based on A Turtle-shell Matrix

A matrix based on turtle shells is constructed before the data hiding process. The matrix only contains 8 integers ranging from 0 to 7. The first row of the matrix is set from 0 to 7 in a circular permutation, and the other rows obey two rules. The first rule is that the difference between two adjacent values in the same row is set to 1. The second rule is that the difference between two adjacent values in the same column is set alternately to 2 and 3. The matrix is shown in Figure 1.

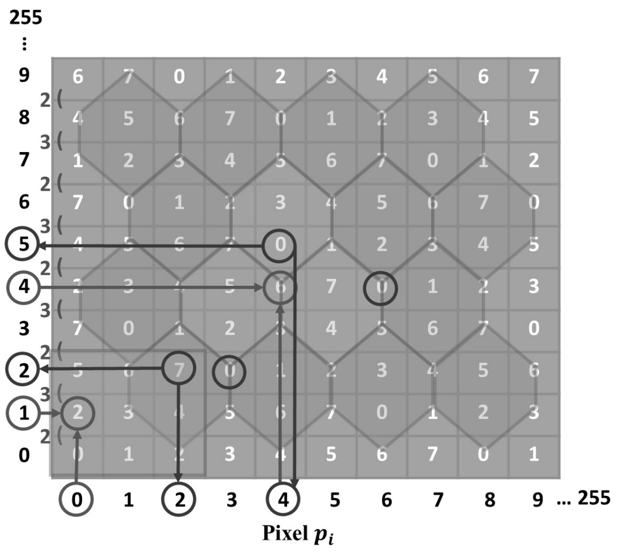


Figure 1: An example of the matrix based on turtle shells

The matrix here is used to hide secret data in order to obtain the minimum distortion between the original image and the stego image. The details of embedding procedure are described as follows.

Step 1. Divide the original image into non-overlapping pixel pairs, each of which contains two adjacent pixels (p_i,p_{i+1}).

Step 2. Generate a random binary bit stream as the secret data and convert each three bits into an octal digit, ranging from 0 to 7.

Step 3. Choose a pixel pair (p_i,p_{i+1}) as the abscissa and ordinate value of the matrix, respectively. Find a corresponding extraction value in the matrix.

Step 4. If the extraction value is equal to the secret digit that is to be embedded, the value of this pixel pair will not be changed. Otherwise, the next step will be implemented.

Step 5. If the extraction value is not equal to the secret digit that is to be embedded, the following two different cases are considered according to the position of the extraction value in the turtle shell.

Case 1: If the extraction value is on the back of the turtle shell, the value that is equal to the secret digit that is to be embedded can be searched within this turtle shell. And the corresponding abscissa and ordinate of this value are used to substitute for the cover pixel pair (p_i,p_{i+1}). However, if the extraction value is on the edge of the turtle shell, the searching range of the value that is equal to the secret digit will be the collection of the turtle shell that contains it.

Case 2: If the extraction value does not involve in any turtle shell, the searching range is a 3×3 sub-block on which this pixel pair is located. Any 3×3 sub-block also contains 8 different values from 0 to 7 because of the architectural property of the matrix.

Step 6. Repeat Steps 3 through 5 until all of the pixel pairs have been processed to embed the secret digit. In this way, a stego image is generated.

For example, suppose that two secret digits, i.e., 0 and 7, are to be embedded into two pixel pairs, i.e., (4, 4) and (0, 1), respectively. Figure 1 shows an example of the matrix based on turtle shells. The corresponding extraction value of pixel pair (4, 4) is 6, which, obviously, it is not equal to the secret digit 0 that is to be embedded. Because this extraction value is on the edge of three turtle shells, the closest 0 is searched within the three turtle shells with the red background in Figure 1. Then the pixel pair (4, 5) is taken as the new pixel pair to substitute for the original pixel pair. In the same way, the extraction value of the pixel pair (0, 1) does not involve in any turtle shell, so the searching range is a 3×3 sub-block with the red background in Figure 1. For this pair, the secret digit is 7 which is located in (2, 2). This pixel pair is replaced by (2, 2).

In the process of extracting secret data, taking each pixel pair of the stego image as a coordinate of the matrix, a corresponding extraction value in the matrix can be determined. It is the embedded secret digit. By converting each extraction value into binary bits in order, the original secret data can be recovered.

2.2 Particle Swarm Optimization

PSO, which was proposed by Kennedy and Eberhart in 1995 [19], is a global search algorithm. It was developed based on the social behavior of flocks of birds and schools of fish when searching for food. This technique has been used successfully in many fields as an optimization tool, such as estimating the distribution state [21] and dispatching reactive power [24].

In PSO, each individual of the population is called a particle, and it flies around in a multi-dimensional search space in order to find the optimal solution. The position of a particle represents a candidate solution to the optimization problem at hand. Each particle searches for a better position in the search space by changing its velocity according to the rules that were inspired originally by behavioral models of flocks of birds. The position of the particle is updated as follows [19]:

$$x_i(t+1) = x_i(t) + v_i(t+1), \tag{1}$$

where $x_i(t+1)$ and $x_i(t)$ are two parameters that represent the updated position and the current position, respectively. The velocity of this particle is updated according to the following equation:

$$\begin{aligned} v_i(t+1) &= wv_i(t) + c_1r_1(p_{best} - x_i(t)) \\ &\quad + c_2r_2(g_{best} - x_i(t)), \end{aligned} \tag{2}$$

where c_1 and c_2 are two acceleration coefficients, w is an inertia factor, r_1 and r_2 are two independent random numbers that are distributed uniformly in the range of [0,1], and p_{best} and g_{best} are the local best particle and the global best particle, respectively.

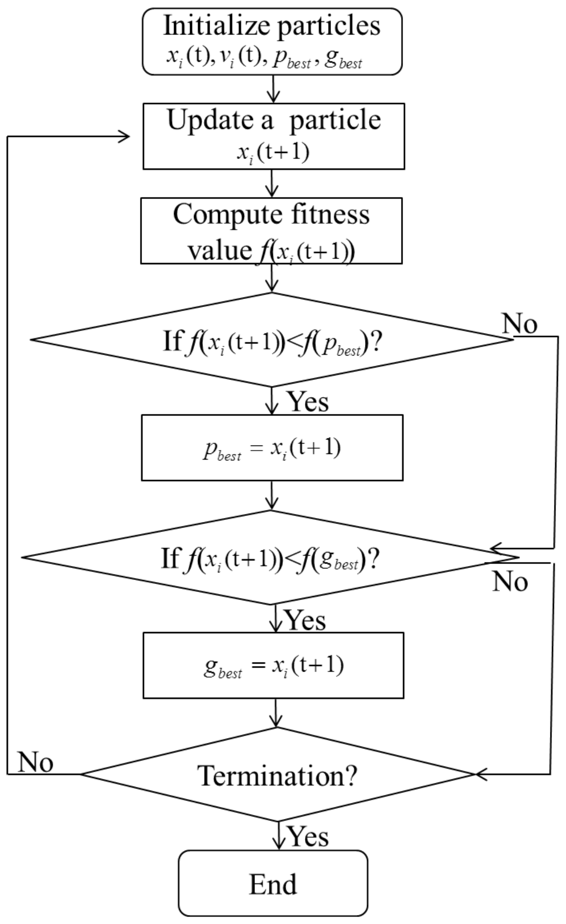

Initialize particles
$x_i(t), v_i(t), p_{best}, g_{best}$
Update a particle
$x_i(t+1)$
Compute fitness value $f(x_i(t+1))$
If $f(x_i(t+1))<f(p_{best})$?
No
Yes
$p_{best} = x_i(t+1)$
If $f(x_i(t+1))<f(g_{best})$?
No
Yes
$g_{best} = x_i(t+1)$
No
Termination?
Yes
End

Figure 2: The flowchart of PSO

Figure 2 shows the basic flowchart of PSO, and the process of PSO is summarized as follows:

Step 1. Initialize a population of particles with random positions $x_i(t)$ and velocities $v_i(t)$ and compute the corresponding fitness values $f(x_i(t))$. Each initial particle is denoted as the local best particle p_{best}, and the particle that has the best fitness value is denoted as the global best particle g_{best}.

Step 2. Update the position of a particle $x_i(t+1)$ according to its updated velocity $v_i(t+1)$, which are determined by using Equations (1) and (2), respectively.

Step 3. Compute the fitness value of the updated particle $f(x_i(t+1))$.

Step 4. Compare this new fitness value $f(x_i(t+1))$ with $f(p_{best})$. If $f(x_i(t+1)) < f(p_{best})$, p_{best} is replaced by $x_i(t+1)$. Otherwise, p_{best} remains unchanged.

Step 5. Compare the current fitness value $f(x_i(t+1))$ with $f(g_{best})$. If $f(x_i(t+1)) < f(g_{best})$, g_{best} is replaced by $x_i(t+1)$. Otherwise, g_{best} remains unchanged.

Step 6. If the terminal condition is met, then the best particle g_{best} and its fitness value are output. Otherwise, go back to Step 2.

3 Proposed Scheme

In the turtle-shell matrix, there are eight integers from 0 to 7 in a turtle shell or a 3×3 sub-block due to the architectural property of the matrix. In order to minimize the distortion of the cover image, it is important to determine the best table to replace the original eight integers. An example of substitution tables is shown in Figure 3. There is a total of 40,320 ways to sort these eight integers in total. So, it would take a lot of time to search all of the tables to find the best one. Thus, in this paper, we used PSO to search for the best table of the matrix.

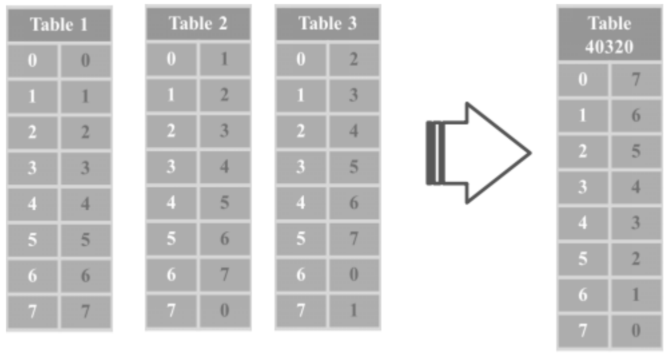


Table 1	
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7

Table 2	
0	1
1	2
2	3
3	4
4	5
5	6
6	7
7	0

Table 3	
0	2
1	3
2	4
3	5
4	6
5	7
6	0
7	1

Table 40320	
0	7
1	6
2	5
3	4
4	3
5	2
6	1
7	0

Figure 3: Substitution tables

3.1 Searching for The Best Replaceable Table Based on PSO

Input: A matrix M based on turtle shells, a cover image CI, secret messages S;

Output: A near optimal replaceable table T;

Step 1. The initial phase.

First, a population of particles is initialized randomly. Each particle represents a table that has 8 different values that range from 0 to 7. Table 1 shows an example of an initial particle, $P_i(t)$=($x_{i1}(t)$,$x_{i2}(t)$,...,$x_{i8}(t)$), where $x_{ij}(t)$ represents the initial position of this particle and each position has a different value. The initial particle denotes its local best particle as p_{best}, and the particle that has the best fitness value is denoted as the global best particle g_{best}. The velocity of each position of a particle, $v_{ij}(t)$, is chosen randomly from 0 through 7.

Step 2. Update one position of a particle.

In order to keep a particle that has 8 different values, each position should be saved before this position is updated. The parameter $temp_{ij}$ is used to save the original values of these particles and it is shown in Equation (3).

$$\begin{aligned} temp_{ij} &= temp_{ij}(t), i = 1, 2, \cdots, n; \\ & \qquad j = 1, 2, \cdots, 8. \end{aligned} \tag{3}$$

The velocity of a particle is updated mainly according to three values, i.e., the local best particle, the global best particle and its particle value, which is shown in Equation (4).

$$\begin{aligned} v_{ij}(t+1) &= wv_{ij}(t) + c_1 r_1 (p_{best(ij)} - x_{ij}(t)) \\ & \quad + c_2 r_2 (g_{best(j)} - x_i(t)), \end{aligned} \tag{4}$$

where i is the i^th particle, j is the j^th position of this particle, c_1 and c_2 are two acceleration coefficients, both of which are set to 1, and w is an inertia factor that is set to 0.4.

Then the particle's position is updated as $x_{ij}(t+1)$ according to its corresponding velocity. When the position of this particle is updated, the corresponding value should be limited in the range of 0 through 7. A modular function is used to achieve this purpose, as shown in Equation (5).

$$v_{ij}(t+1) = (x_{ij}(t)) + v_{ij}(t+1)) \bmod 8. \tag{5}$$

If the new value has already existed in this particle, the old value will be replaced by the saved value $temp_{ij}$ that is shown in Equation (6). In other words, the two values of this particle in different positions are exchanged. In each circulation, only one position

Table 1: An example of an initial particle $P_i(t)$

position	$x_{i1}(t)$	$x_{i2}(t)$	$x_{i3}(t)$	$x_{i4}(t)$	$x_{i5}(t)$	$x_{i6}(t)$	$x_{i7}(t)$	$x_{i8}(t)$
value	2	5	7	1	6	3	4	0

of the particle is updated. In this way, this particle is updated as $P_{ij}(t+1)$=($x_{i1}(t)$,$x_{i2}(t)$,...,$x_{ij}(t+1)$,...,,$x_{i8}(t)$).

$$x_{ik}(t+1) = \begin{cases} temp_{ij} & \text{if } x_{ij}(t+1) = x_{ik}(t) \\ & k = 1, 2, \ldots, 8, k \neq j, \\ x_{ik}(t) & \text{otherwise} \end{cases} \quad (6)$$

Step 3. Compute the corresponding fitness value.

After getting the new updated particle, $P_i(t+1)$, it is used to replace the eight integers in the matrix that is from 0 through 7. Then, a new matrix is generated, and it is used to embed secret data into cover images. In order to judge whether the new matrix is better than the original matrix, the fitness function is used to measure the distortion between the cover image and the stego-image as shown in Equation (7).

$$f(P_i(t+1)) = \sum_{i=1}^{H} \sum_{j=1}^{W} (CI_{hw} - SI_{hw})^2, \quad (7)$$

where CI_{hw} and SI_{hw} are the pixel values of the original image and the cover image, respectively, and $H \times W$ is the size of the image.

For example, if we use the particle, $P_i(t)$ = (2, 5, 7, 1, 6, 3, 4, 0), in Table 1 to replace the original eight integers, $P_0(t) = (0, 1, 2, 3, 4, 5, 6, 7)$, one by one at corresponding positions, a new matrix will be obtained, as shown in Figure 4. The main architectural property of the matrix is not changed, which means that any 3×3 sub-block or a turtle shell still contains 8 different values from 0 through 7. Suppose that two secret digits, e.g., 0 and 7, must be embedded into two pixel pairs, i.e., (4, 4) and (0, 1), respectively. According to the theory mentioned in Subsection 2.1, the new pixel pairs will be (5, 4) and (0, 1) after embedding the secret digit based on the new matrix. The fitness value between the original two pixel pairs and the new pixel pairs is computed as $f(P_i(t))$=$(4-4)^2$+$(5-4)^2$+$(4-4)^2$+$(0-0)^2$+$(1-1)^2$ based on the new matrix. If the original matrix is used to hide the two secret digits by using the same pixel pairs, as shown in Figure 1, the new pixel pairs will be (4, 5) and (2, 2). The corresponding fitness value is computed as $f(P_0(t))$=$(4-4)^2 + (5-4)^2 + (4-4)^2 + (2-0)^2 + (2-1)^2$. Since $f(P_i(t))$¡$f(P_0(t))$, the distortion associated with embedding the secret digits using the new matrix is less than that by using the original matrix.

Step 4. Update p_{best} and g_{best}.

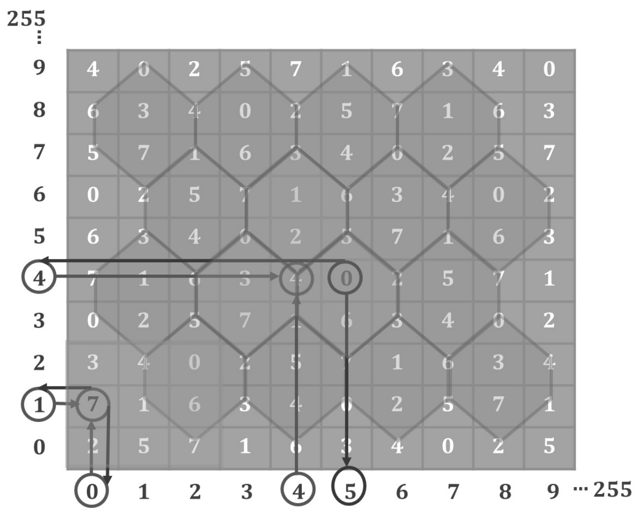


Figure 4: An example of the new matrix based on turtle shells

If $f(P_i(t+1))$¡$f(p_{best})$, p_{best} will be updated by this new particle. Otherwise, p_{best} will remain unchanged. If $f(P_i(t+1))$¡$f(g_{best})$, g_{best} will be updated by this new particle. Otherwise, g_{best} will remain unchanged.

Step 5. Termination.

If the terminal condition is encountered, then the best particle, g_{best}, is output. Otherwise, go back to Step 2. Here, the terminal condition is the maximum iteration time. After the near optimal table T is obtained, it is saved as a secret key for hiding and extracting the secret data.

3.2 Data Hiding Procedure

In this procedure, first, a new matrix is generated by replacing the original eight integers ranging from 0 through 7 in ascending order with the integers in the near optimal table. And then, each three bits of the secret messages are converted into a decimal digit ranging from 0 to 7. Selecting a pair of pixels from a cover image CI as a pair of coordinates of the new matrix, the secret digit is embedded in the same way that is used in the turtle shell scheme. After all of the pixel pairs are used for embedding secret digits, a stego-image SI is generated, and it is delivered, along with the near optimal table T, to a participant.

3.3 Data Extracting Procedure

In this procedure, first, the original matrix is generated using the original rule described in related work. By using the saved near optimal table T, a new matrix can be generated by replacing the original eight integers ranging from 0 to 7 in an ascending order with the integers in the best table. Then select a pair of pixels from the stego-image SI as a coordinates of the new matrix for extracting the secret digits. After all of the stego pixel pairs have been mapped into the new matrix, the original binary secret messages S can be restored from the extracted digits.

4 Experiments and Discussion

This section describes the experiments that were conducted on a group of 512×512 gray-level images that are shown in Figure 5. The simulation environment of our experiments was a PC with a 2.1 GHz CPU and 4 GB RAM. All schemes were implemented by using MATLAB 2013a. The population of particles was set to 8, and the maximum iteration times were set to 20 in our algorithm.

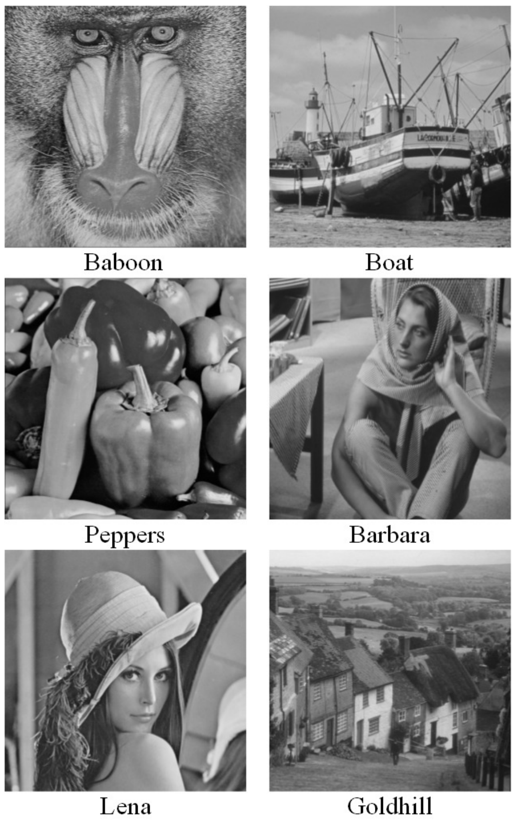


Figure 5: Six 515×512 gray images

The $PSNR$ was used to evaluate the visual quality of the image after embedding the secret data, which is shown in Equation (8). O_{hw} and C_{hw} denote the pixel values of the original image and the cover image, respectively, and $H \times W$ denotes the size of the image.

$$PSNR = 10log_{10}(\frac{255^2}{MSE}), \quad (8)$$

where

$$MSE = \frac{1}{H \times W}\sum_{h=1}^{H}\sum_{w=1}^{W}(O_{hw} - C_{hw})^2. \quad (9)$$

In order to prove the efficiency of our algorithm, we used the exhaustive search and a genetic algorithm (GA) to search the best table. The exhaustive search was intended to identify the best table among the 40,320 tables. For the GA, a chromosome is presented as a table that contains 8 integers ranging from 0 to 7 and each integer represents a gene of this chromosome. A chromosome was selected according to its probability, which depended on the corresponding $PSNR$ value. The population size was set to 8. The crossover operator was implemented by exchanging four genes of two chromosomes. The probability of crossover was controlled by a parameter that was set to 0.8. The mutation operation was to exchange two genes of one chromosome and the probability of mutation was set to 0.001.

We used the scheme based on the turtle-shell matrix to hide the secret digits. There are 393,216 bits embedded into the cover image and the embedding capacity (EC) is 18.75% of the size of the cover image. Table 2 provides a comparison of various schemes. Three different methods were used to search the best table to obtain the minimum distortion of the turtle-shell matrix, i.e., the exhaustive search, GA, and PSO. From Table 2, it is apparent that exhaustive search had the best $PSNR$ values among the three schemes. Both GA and our scheme achieved near optimal results, and the average improvements of $PSNR$ values for the two schemes were 0.02 and 0.03dB, respectively, compared with the original scheme based on the turtle-shell matrix.

The main purpose of our scheme is to find a best table to obtain the minimum distortion of the turtle-shell matrix. Recently, Liu et al. [18] enhanced the embedding capacity by using the location table to develop the turtle shell-based scheme. However, the turtle-shell matrix used in Liu et al.'s scheme is also not optimal, meaning that our scheme can be used in [18] to obtain the minimum distortion of the turtle-shell matrix. The experimental results are shown in Table 3. There are 524,288 bits embedded into the cover image and the embedding capacity (EC) is up to 25% of the size of the cover image. From Table 3, it is obvious that our scheme achieves higher $PSNR$ values.

Table 4 shows the encoding time requited of different schemes. Although [7] and [18] cost less time than other schemes, the turtle-shell matrix in their schemes is not optimal. The exhaustive search took the most time to find the best table among these methods, because all of the tables were tested. The GA took less time than the exhaustive search, but the $PSNR$ values were lower than those for the exhaustive search and PSO, as shown in

Table 2: Comparison of various schemes

Images	**Ref.[18]**		**Exhaustive search**		**GA**		**PSO**	
	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR
Baboon	18.75%	49.45	18.75%	49.49	18.75%	49.48	18.75%	49.49
Boat	18.75%	49.46	18.75%	49.48	18.75%	49.48	18.75%	49.48
Peppers	18.75%	49.44	18.75%	49.49	18.75%	49.48	18.75%	49.49
Barb	18.75%	49.45	18.75%	49.49	18.75%	49.48	18.75%	49.49
Lena	18.75%	49.48	18.75%	49.51	18.75%	49.49	18.75%	49.50
Goldhill	18.75%	49.46	18.75%	49.49	18.75%	49.47	18.75%	49.48
Average	18.75%	49.46	18.75%	49.49	18.75%	49.48	18.75%	49.49

Table 3: Comparisons of the proposed scheme and [18]

Images	**Ref.[19]**		**PSO**	
	EC	PSNR	EC	PSNR
Baboon	25%	45.55	25%	45.57
Boat	25%	45.55	25%	45.58
Peppers	25%	45.54	25%	45.56
Barb	25%	45.56	25%	45.58
Lena	25%	45.55	25%	45.57
Goldhill	25%	45.49	25%	45.52
Average	25%	45.54	25%	45.56

Table 2. Compared with other methods, our PSO algorithm achieved near optimal results and more efficiently identified the near optimal table.

In addition, the near optimal table can be used as a secret key for the extraction of secret data. There were 40,320 tables to generate a matrix for the extraction of data extraction. If a receiver were to use a fake table to generate the matrix of turtle shells, the false secret data would be extracted by using the pixel pairs of the stego-image. Therefore, the security of the scheme based on the turtle-shell matrix was enhanced by using our method.

5 Conclusions

In this paper, we proposed a novel scheme to optimize data hiding based on turtle shells. First, we generated an original matrix based on the turtle shell scheme. Then PSO was used to search the near optimal table of the matrix. The integers in the matrix of the turtle shells are replaced by the near optimal table in order to minimize the distortion of the image. Two pixels of the cover image were used to embed secret digits according to this near optimal matrix. Our scheme has the same embedding capacity as the original data hiding scheme based on turtle shells, but it provides higher PSNR values. Also, the near optimal table can be used as a secret key for extracting data, which enhances the security of the scheme based on the turtle-shell matrix. The experimental results showed that the proposed scheme had better visual quality than the original scheme based on the turtle-shell matrix and was efficient in finding the near optimal table.

Acknowledgments

This work has been supported in part by National Natural Science Foundation of China (No. 61272262 and No. 61210006), The Program of "One hundred Talented People" of Shanxi Province, Research Project Supported by Shanxi Scholarship Council of China (2014-056), Program for New Century Excellent Talent in Universities (NCET-12-1037), International Cooperative Program of Shanxi Province (No. 2015081015), and Scientific and Technological project of Shanxi Province (2015031003-2). Innovative Projects for Graduate Students of TYUST (No.20145018).

References

[1] A. A. Abdelwahab and L. A. Hassan, "A discrete wavelet transform based technique for image data hiding," in *Proceedings of 25th National Radio Science Conference*, pp. 1–9, Egypt, 2008.

[2] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," *IBM Systems Journal*, vol. 35, no. 3&4, pp. 313–336, 1996.

[3] N. Bourbakis and C. Alexopoulos, "Picture data encryption using scan patterns," *Pattern Recognition*, vol. 25, no. 6, pp. 567–581, 1992.

[4] C. C. Chang, Y. C. Chou, and T. D. Kieu, "An Information Hiding Scheme Using Sudoku," in *Proceedings of the Third International Conference on Innovative Computing, Information and Control*, pp. 17–22, 2008.

[5] C. C Chang, T. D. Kieu, and W. C. Wu, "A lossless data embedding technique by joint neighboring coding," *Patter Recognition*, vol. 42, no. 7, pp. 1597–1603, 2009.

[6] C. C. Chang, C. C. Lin, C. S. Tseng, and W. L. Tai, "Reversible hiding in DCT-based compressed images," *Information Sciences*, vol. 177, no. 13, pp. 2768–2786, 2007.

Table 4: Encoding time requited of different schemes (s)

Images	**Ref.[18]**	**Ref.[19]**	**Exhaustive search**	**GA**	**PSO**
Baboon	12.35	25.47	54149.21	1284.19	175.21
Boat	13.14	24.52	59712.17	1230.43	171.74
Peppers	11.86	23.76	54038.28	1238.56	173.36
Barb	11.57	24.15	52824.63	1142.42	173.15
Lena	12.63	25.24	54856.31	1190.71	172.97
Goldhill	13.72	24.68	59942.15	1302.48	173.63
Average	12.55	24.64	55920.46	1231.47	173.34

[7] C. C. Chang, Y. Liu and T. S. Nguyen, "A novel turtle shell based scheme for data hiding," in *Proceedings of 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, pp. 89–93, 2014

[8] C. C. Chang and W. C. Wu, "Hiding secret data adaptively in vector quantisation index tables," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 153, no. 5, pp. 589–597, 2006.

[9] W. C. Du, and W. J. Hsu, "Adaptive data hiding based on VQ compressed images," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 150, no. 4, pp. 233–238, 2003.

[10] W. Hong, T. S. Chen, and C. W. Shiu, "A Minimal Euclidean Distance Searching Technique for Sudoku Steganography," in *Proceedings of International Symposium on Information Science and Engineering*, vol. 1, pp. 515–518, 2008.

[11] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched Pareto genetic algorithm for multiobjective optimization," in *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, vol. 1, pp. 82–87, 1994.

[12] Y. C. Hu, "High capacity image hiding scheme based on vector quantization," *Pattern Recognition*, vol. 39, no. 9, pp. 1715–1724, 2006.

[13] J. K. Jan, and Y. M. Tseng, "On the security of image encryption method," *Information Processing Letters*, vol. 60, no. 5, pp. 261–265, 1996.

[14] J. Kennedy, and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of ?the 1995 IEEE international conference on neural network*, pp. 1942–1948, 1995.

[15] H. J. Kim, C. Kim, Y. Choi, S. Wang, and X. Zhang, "Improved Modification Direction Schemes," *Computers & Mathematics with Applications*, vol. 60, pp. 319–325, 2010.

[16] C. C Lin, and P. F. Shiu, "High capacity data hiding scheme for DCT-based images," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 3, pp. 220–240, 2010.

[17] H. Liu, J. Liu, J. Huang, D. Huang, and Y. Q. Shi, "A robust DWT-based blind data hiding algorithm," in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 672–675, Arizona, USA, 2002.

[18] Y. Liu, C. C. Chang, and T. S. Nguyen, "High capacity turtle shell-based data hiding," *IET Image Processing*, vol. 10, no. 2, pp. 130–137, 2016.

[19] B. Liu, L. Wang, and Y. H. Jin, "An effective PSO-Based Memetic Algorithm for Flow Shop Scheduling," *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, vol. 37, no. 1, pp. 18–27, 2007.

[20] J. Mielikainen, "LSB matching revisited," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 285–287, 2006.

[21] S. Naka, T. Genji, and T. Yura, "A hybrid particle swarm optimization for distribution state estimation," *IEEE Transactions on Power Systems*, vol. 18, no 1, pp. 60–68, 2003.

[22] R. Z. Wang, C. F. Lin, and J. C. Lin, "Image hiding by optimal LSB substitution and genetic algorithm," *Pattern Recognition*, vol. 34, no. 3, pp. 671–683, 2001.

[23] X. Zhang, and S. Wang, "Efficient Steganographic Embedding by Exploiting Modification Direction," *IEEE Communications Letters*, vol. 10, no. 11, pp. 781–783, 2006.

[24] B. Zhao, C. X. Guo, and Y. J. Cao, "A multiagent-based particle swarm optimization approach for optimal reactive power dispatch," *IEEE Transactions on Power Systems*, vol. 20, no. 2, pp. 1070–1078, 2005.

Qiang Jin was born in Shanxi Province, China, in 1989. He is currently pursuing the M.E. degree in Taiyuan University of Science and Technology. His current research interests include data hiding, and image compression.

Zhihong Li is currently an associate professor in Taiyuan University of Science and Technology, China. He received the B.Eng. in electronic information engineering from Taiyuan University of Science and Technology in 1994. His research interests include compressed sensing, and secret image sharing. He was participated in the projects on distributed video coding and now is leading one research project on image secret from Shanxi Natural Science Foundation.

Chin-Chen Chang received the B.S. degree in applied mathematics and the M.S. degree in computer and decision sciences from National Tsing Hua University, Hsinchu, Taiwan, R.O.C., in 1977 and 1979, respectively. He received his Ph. D in computer engineering in 1982 from the National Chiao Tung University, Hsinchu, Taiwan. Since February 2005, he has been a Chair Professor of Feng Chia University. In addition, he has served as a consultant to several research institutes and government departments. His current research interests include database design, computer cryptography, image compression and data structures.

Anhong Wang received B.E and M.E. degrees from Taiyuan University of Science and Technology (TYUST) respectively in 1994 and 2002, and Ph. D degree in Institute of Information Science, Beijing Jiaotong University in 2009. She became an associate professor with TYUST in 2005 and became a professor in 2009. She is now the director of Institute of Digital Media and Communication, Taiyuan University of Science and Technology. Her research interest includes image/video coding, compressed sensing, and secret image sharing. Now she is leading two national research projects from National Science Foundation of China.

Li Liu received her B.E. degree in communication engineering in 2002, from Lanzhou Railway University and M.E. degree in communication and information system in 2006, from Lanzhou Jiaotong University. Her current research interests include image compression and secret sharing.

Social Bots Detection on Mobile Social Networks

Cheng Binlin[1], Fu Jianming[2]
(Corresponding author: Cheng Binlin)

School of Computer Science, Wuhan University[1]
Wuhan 430079, Hubei, P.R. China
School of Computer Science, Wuhan University[2]
Wuhan 430079, Hubei, P.R. China
(Email: binlincheng@163.com)

(Received Dec. 30, 2015; revised and accepted Mar. 11 & Apr. 9, 2016)

Abstract

Mobile social networks have become very popular in recent years. The popularity of mobile social networks has attracted a large number of companies to do marketing on it. However, the social marketing suffer from social bots, a kind of bot accounts. In this paper, SBDSF(social bots detection based on the number of shared friends), a social graph based approach is proposed to detect social bots . SBDSF use the feature of social graph to detect social bots. We measure the effectiveness of SBDSF, the result of evaluation shows that SBDSF can achieve 96.1% accuracy and 95.1% precision using the Neural Network classifier.

Keywords: Mobile social networks, security, social bots

1 Introduction

Mobile social networks have become a trend in this modern time. It spreads rapidly attracting a lot of new users [4, 5].

As mobile social networks becomes increasingly popular in the world, The popularity of mobile social networks has attracted a large number of companies to do marketing on it. However, the social marketing suffer from social bots, a kind of bot accounts [3, 6, 8].

As the bot accounts registered by automatic program, social bots have many profile features different with real human accounts. For Example, social bots have less followers and post less tweets. There are some applications on Mobile social networks attempt to detect social bots by these profile features. To evade detection, the social bots have evolved from low-lever social bots to high-lever social bots . High-lever social bots have a certain amount of followers, publish tweets every day, the profile feature based approach can't detect the high-lever social bots.

As the profile features are easy to alter, we aim to answer the question: can we design a social bots detection approach which not relies on the profile feature? Some users announced that they no more used the Mobile social networks as they don't like the social bots , if the Mobile social networks provider can detect social bots in its system effectively, it can improve the experience of its users and attract more companies to do the Mobile social networks marketing in the Mobile social networks.

We propose SBDSF (social bots detection approach based on the number of Shared Friends), a social graph based social bots approach. SBDSF does not rely on the profile features of Mobile social networks account; instead, our approach is focused on the social graph structure.

In the paper, we plan to give the formal description of Shared Friends feature, and propose a social graph based social bots approach, then we validate the efficacy of the classification system based on the feature of the number of Shared Friends.

The rest of the paper is organized as follows. Section 2 introduces the goals of our system. Section 3 gives the approach our system. Section 4 presents our experimental classification results of social bots on Wechat. Section 5 concludes.

2 Goals

SBDSF aims to identify social bots accounts using the social graph based feature. Our design has the following main goals:

1) Effectiveness.
The approach should identify mostly social bots (low false positives), while limiting the number of normal followers considered social bots (low false negatives).

2) Efficiency.
The system should have a low performance overhead. It should be feasible to deploy to handle large mobile social networks by Microblog providers in practice.

3) Robustness.
The approach should be robust under various at-

tack strategies, and be immune to sophisticated evasion techniques. We assume that the attacker knows about the SBDSF techniques and will try to avoid detection.

3 Approach

In this section, we describe the basic idea of social bots detection, formalize the problem of social bots detection.

Definition 1. ***Follower*** *node v_j is a follower of node v_i if the edge $a = (j, i)$ is contained in the set of edges, followers are the incoming links of a node.*

Definition 2. ***Friend*** *node v_j is a follower of node v_i if the edge $a = (i, j)$ is contained in the set of edges, followers are the incoming links of a node.*

Definition 3. ***Shared Friend.*** *In the Weibo social graph, we use set $F(i)$ denoting the vertex which is the friend of vertex$_i$: $F(i) = \{ j \mid \{vertex_j \text{ is the friend of } vertex_i \}$; then the common friends of vertex$_i$ and vertex$_j$ can be defined as:*

$$C(i, j) = F(i) \cap F(j). \tag{1}$$

Definition 4. ***Shared Friends Graph.*** *For a Weibo social graph $G(V, E)$, we define Shared Friends Graph as $G_C = (V_C, E_C)$. The V_C is the vertices set which is identical to the V of graph $G(V, E)$. An weighted edge $E_C = (i, j)$ is linking V_{C_i} and V_{C_j}, which stands for there are Shared Friends between V_i and V_j, and the weights of E_C is equal to the number of Shared Friends of V_i and V_j. Given a $E_C = (i, j)$, there is a equation shown as follow:*

$$Weights\ (E_C) = C(i, j). \tag{2}$$

In a summary, social graph $G(V, E)$ is unweighed directed graph, while Shared Friends Graph $G_C(V_C, E_C)$ is weighted undirected graph.

Example 1. *Consider a social graph $G(V, E)$ in Figure 1, the corresponding Shared Friends Graph is shown as G_C (V_C, E_C) in Figure 2. User b and d has a common friend which is User e; User c and d has two common friend User a and e.*

In order to see the difference of Shared Friends Graph on zombie friends and normal ones intuitively, we generate the Shared Friends Graph on the sample of normal Friends and zombie friends respectively. Figure 3 is the example of Shared Friends Graph on normal friends Sample, while Figure 4 is the example of Shared Friends Graph on zombie friends Sample. We can find that the number of Shared Friends among zombie followers is greater than that of normal ones significantly.

Definition 5. ***Community Aggregation Degree.*** *The Community Aggregation Degree denotes the degree of the community aggregation:*

$$D_{community} = E_{community}/E_{total} \tag{3}$$

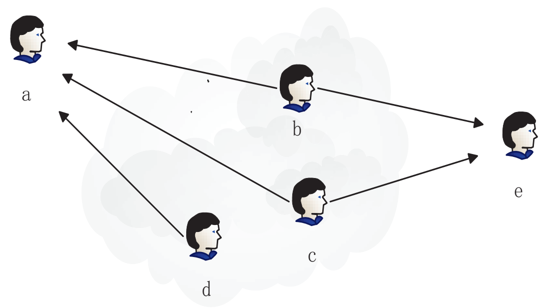


Figure 1: Social graph

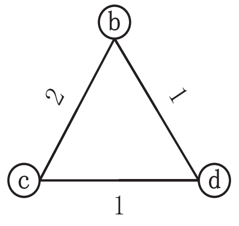


Figure 2: Shared friends graph

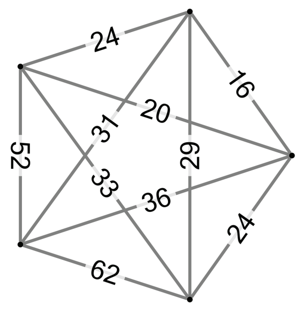


Figure 3: Sample of normal account

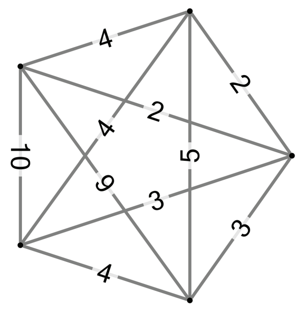


Figure 4: Sample of social bots

The $D_{community}$ denotes Community Aggregation Degree of the community, $E_{community}$ denotes the number of the edges in the Community, E_{total} denotes the number of the edges of the vertices in the Community.

We use the Community Aggregation Degree of the Shared Friends Graph the detect the social bots in this paper.

4 Evaluation

This section discusses the evaluation of our system.

4.1 Datasets

Wechat is the top mobile social networks in China, it is known to all that Twitter and Facebook are two popular ones in American and Erope while Wechat is the most widely used application in China [1]. Wechat offers API [7, 2], and we use it to crawl and collect data.

The sample set contains two subsets (Table 1): benign set and bots set. Benign set consist of 5,000 known, benign crawled ID from 10 seed ID which is randomly select, the bots set consist of 5,000 social bots which we purchased from 5 different sellers on Taobao site (Chinese version of Ebay).

Table 1: Evaluation dataset

Set	Source
5,000 benign accounts	crawled from Wechat
5,000 social bots	purchased on Taobao site

4.2 Evaluation Result

To increase the difficulty of detection, we mix randomly with the Benign set and bots set to build the Follower set. In our evaluation, bots set consist of 5,000 samples; the ratio of social bots account for all the followers is 40%, which is quite a low ratio in practice but nevertheless presents no problems for social bots detection.

We use the classifiers based on the Community Aggregation Degree of Community Aggregation Degree to detect the social bots. All the classifiers reported in the evaluation are computed using 10-fold cross validation. Table 2shows the evaluation result of different classification algorithms.

Table 2: Evaluation dataset

Classifier	Accuracy	Precision
Decision Tree	92.1%	93.1%
Neural Network	96.1%	95.1%
Support Vector Machines	95.0%	94.1%
Native Bayesian	90.3%	88.6%

It can be seen that Neural Network classifier has the best overall performance compared with other algorithms.

5 Conclusions

The popularity of mobile social networks makes it being a great platform to do marketing. Social bots become the major threat of social marketing. The social bots is in evolution, previous work can detect low-lever social bots but not high-lever social bots. To this end, we have proposed SBDSF, a social graph based approach to detect social bots, which use the feature that the number of shared friends among the social bots from a purchaser is usually greater than that of the normal users.

The popularity of mobile social networks makes it being a great platform to do marketing. Social bots become the major threat of social marketing. The social bots is in evolution, previous work can detect low-lever social bots but not high-lever social bots. To this end, we have proposed SBDSF, a social graph based approach to detect social bots, which use the feature that the number of shared friends among the social bots from a purchaser is usually greater than that of the normal users.

References

[1] F. Gao and Y. Zhang, "Analysis of wechat on iphone," in *2nd International Symposium on Computer, Communication, Control and Automation*, vol. 69, pp. 278–281, 2013.

[2] C. H. Lien and Y. Cao, "Examining wechat users' motivations, trust, attitudes, and positive word-of-mouth: Evidence from china," *Computers in Human Behavior*, vol. 41, pp. 104–111, 2014.

[3] A. Muthana, A. A. A. Ghani, and R. Mahmod, "It cannot get away: An approach to enhance security of user account in online social networks," *International Journal of Computer Science and Network Security*, vol. 15, no. 4, pp. 1, 2015.

[4] V. Nincic, M. Chang, and F. Lin, "Dmlrid-an xml-based proof-of-concept mobiledrm framework for sharing learning contents amongmobile networks," *International Journal of Information and Electronics Engineering*, vol. 3, no. 2, pp. 180, 2013.

[5] M. Romoozi and H. Babaei, "A P2P fuzzy reputation system based on security policies for mobile ad-hoc networks," *International Journal of Information and Electronics Engineering*, vol. 3, no. 5, pp. 481, 2013.

[6] S. Sarpong, C. Xu, and X. Zhang, "An authenticated privacy-preserving attribute matchmaking protocol for mobile social networks," *International Journal of Network Security*, vol. 17, no. 3, pp. 357–364, 2015.

[7] L. Wu and X. Yang, "A research on the services of university mobile library based on the wechat public platform [j]," *Research on Library Science*, vol. 18, pp. 013, 2013.

[8] P. Wuttidittachotti and T. Daengsi, "Quality evaluation of mobile networks using voip applications: A case study with skype and line based-on stationary tests in bangkok," *International Journal of Computer*

Network and Information Security, vol. 7, no. 12, pp. 28, 2015.

Cheng Binlin received his M.S degree from Wuhan University in 2009, China. Currently he is a doctor candidate in Wuhan University. Student member of China Computer Federation. His main research interests include software security and network security.

Fu Jianming received his Ph.D. degree from Wuhan University in 2000, China. Now he is a professor in Wuhan University. Senior member of China Computer Federation. His main research interests include software security and network security.

Guide for Authors
International Journal of Network Security

IJNS will be committed to the timely publication of very high-quality, peer-reviewed, original papers that advance the state-of-the art and applications of network security. Topics will include, but not be limited to, the following: Biometric Security, Communications and Networks Security, Cryptography, Database Security, Electronic Commerce Security, Multimedia Security, System Security, etc.

1. Submission Procedure

Authors are strongly encouraged to submit their papers electronically by using online manuscript submission at http://ijns.jalaxy.com.tw/.

2. General

Articles must be written in good English. Submission of an article implies that the work described has not been published previously, that it is not under consideration for publication elsewhere. It will not be published elsewhere in the same form, in English or in any other language, without the written consent of the Publisher.

2.1 Length Limitation:

All papers should be concisely written and be no longer than 30 double-spaced pages (12-point font, approximately 26 lines/page) including figures.

2.2 Title page

The title page should contain the article title, author(s) names and affiliations, address, an abstract not exceeding 100 words, and a list of three to five keywords.

2.3 Corresponding author

Clearly indicate who is willing to handle correspondence at all stages of refereeing and publication. Ensure that telephone and fax numbers (with country and area code) are provided in addition to the e-mail address and the complete postal address.

2.4 References

References should be listed alphabetically, in the same way as follows:
For a paper in a journal: M. S. Hwang, C. C. Chang, and K. F. Hwang, ``An ElGamal-like cryptosystem for enciphering large messages," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 2, pp. 445--446, 2002.

For a book: Dorothy E. R. Denning, *Cryptography and Data Security*. Massachusetts: Addison-Wesley, 1982.

For a paper in a proceeding: M. S. Hwang, C. C. Lee, and Y. L. Tang, ``Two simple batch verifying multiple digital signatures," in *The Third International Conference on Information and Communication Security (ICICS2001)*, pp. 13--16, Xian, China, 2001.

In text, references should be indicated by [number].

Subscription Information

Individual subscriptions to IJNS are available at the annual rate of US$ 200.00 or NT 7,000 (Taiwan). The rate is US$1000.00 or NT 30,000 (Taiwan) for institutional subscriptions. Price includes surface postage, packing and handling charges worldwide. Please make your payment payable to "Jalaxy Technique Co., LTD." For detailed information, please refer to http://ijns.jalaxy.com.tw or Email to ijns.publishing@gmail.com.