3-6 Toward Realizing Privacy-Preserving IP-Traceback

NOJIMA Ryo

The IP-traceback technology enables us to trace widely spread illegal users on Internet. However, to deploy this attractive technology, some problems have been remained unsolved. One of the biggest issues among them is the *privacy* problem. That is, there is a possibility of tracing not only the illegal users but also the *legal* ones.

In this paper, we show, by using the modern cryptography, the solution to the above problem. Especially, the effectiveness of our oblivious symmetric encryption to the privacypreserving IP-traceback is introduced.

Keywords
IP-traceback, Privacy-preserving, Oblivious symmetric encryption

1 Introduction

1.1 Background

With the rapid advancement of the Internet, network security issues such as computer viruses and DOS attacks have become major concerns in recent years. Among these issues, we have been specifically focusing on **IP-traceback technology** that is used to trace illegal users who launch DOS attacks. Although IP-traceback technology is considered to be useful, it could disclose the privacy of not only illegal users, but also legitimate users. Therefore, we have been also studying/developing IP-traceback technology that can preserve privacy.

The issues related to IP-traceback and privacy-preserving IP-traceback can be simplified as follows. First, consider two users, Alice and Bob. Alice possesses the set of IP addresses $A = \{a_1, \ldots, a_n\}$, and Bob possesses an IP address a. Bob's purpose is to find out whether a is included in A. This problem can be solved when Bob sends a to Alice and she checks whether a is included in A. In fact, the similar process is performed in IP-traceback. On the other hand, things are slightly more complicated in privacy-preserving IP-traceback. In order to realize this technology, it is necessary to check whether a is included in A without disclosing Alice's A and Bob's a. This problem seems impossible to solve, however, we have managed to do so by developing and applying technology called oblivious symmetric encryption. This paper introduces this technology.

1.2 Related research

The aforementioned problem is a specific case of the **secure set-intersection problem** (Fig. 1), which has been widely discussed in the field of cryptographic protocol research. It can be described, in a similar way to **1.1**, as follows:

> Alice and Bob possess a secret set of S_A and S_B, respectively. They want to know only the intersection of S_A and S_B. However, they do not want the other person to know the other elements in their own sets.

For example, let $S_A = \{1, 345, 787, 88\}$, $S_B = \{9893, 3232, 89, 345\}$. Now if we know that $S_A \cap S_B = \{345\}$, we therefore need to make only $S_A \cap S_B = \{345\}$ available to be obtained without disclosing Alice's $\{1, 787, 88\}$ and Bob's $\{9893, 3232, 89\}$. Therefore, it can be said that this problem is a generalized version of the problem we want to solve.

The general solution for this problem has been proposed by Freedman et al.[1].

2 Existing method

2.1 Public-key encryption scheme that is homomorphic with respect to addition

We start with public-key encryption scheme (Fig. 2) that is homomorphic with respect to addition, in order to introduce the related research.

In the public-key encryption scheme, the key for encryption (public key) *pk* differs from the key for decryption (secret key) *sk*. The user (recipient) who possesses the secret key *sk* only discloses *pk*. The sender encrypts the message by using *pk* before sending it to the recipient. The recipient decrypts the ciphertext by using *sk* to obtain the message. Here the ciphertext of the message *m* is expressed as Enc (m) or Enc (pk, m). It is possible to obtain Enc (m_1+m_2) from Enc (m_1), Enc (m_2) without the secret key *sk* in the homomorphic cryptosystem. This type of cryptosystem includes Paillier cryptosystem[2] and ElGamal cryptosystem.

2.2 Configuration in the existing research

First, this paper will discuss the configuration method of set-intersection protocol (not obfuscated), then obfuscate the method.

Assume the universal set $U(|U|=N)$, and consider the vector representation of its subset. That is, let S denote the subset of the universal set U, and V a vector of length N, and define if $x \in S$ then $V[x-1]=1$, and if $x \notin S$ then $V[x-1]=0$. For example, if $U=\{1, 2, 3, 4, 5\}$, $S=\{1, 3, 5\}$, then V, the vector representation

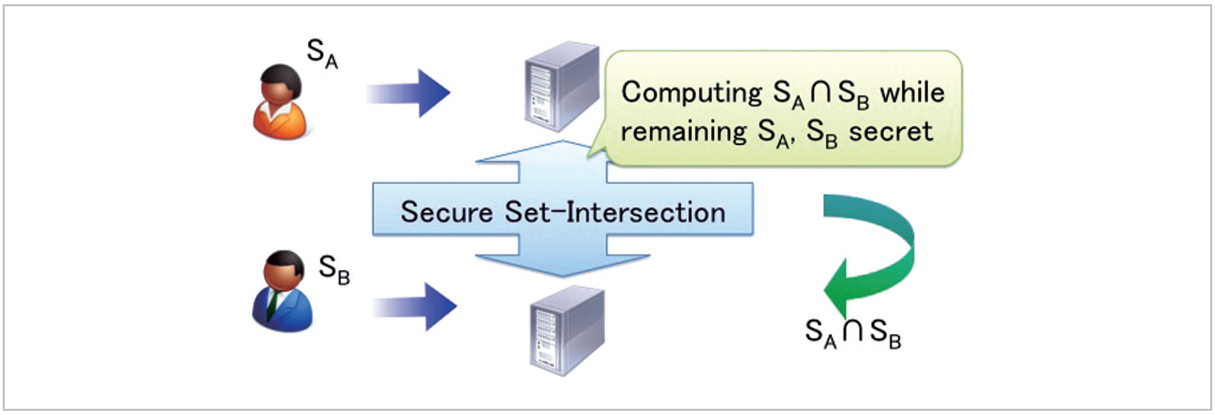
S_A
Computing $S_A \cap S_B$ while remaining S_A, S_B secret
Secure Set-Intersection
S_B
$S_A \cap S_B$

Fig.1 *Secure set-intersection problem*

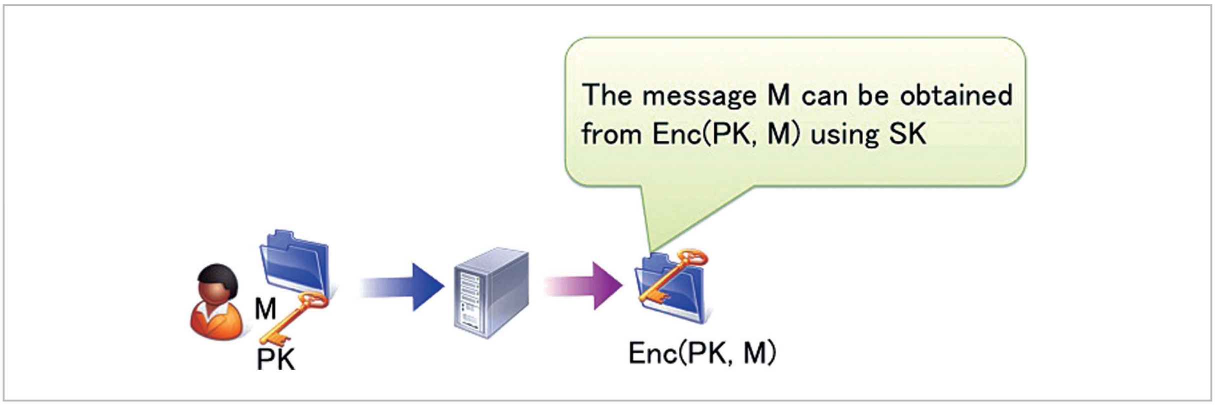

Fig.2 *Description of public-key encryption*

of S, is given by $V=[1, 0, 1, 0, 1]$.

Method 1

Input: Alice's input is S_A, Bob's input is S_B

Step 1: Alice converts S_A to a vector V_A, then sends it to Bob.

Step 2: Bob outputs the intersection of V_A and S_B.

Use a homomorphic cryptosystem to obfuscate the method 1.

Secure set-intersection protocol (method 1)

Input: Alice's input is S_A (V_A), pk,
Bob's input is S_B (V_B), pk, sk.

Step 1: Bob sends $\mathrm{Enc}(V_B[0])$, $\mathrm{Enc}(V_B[1])$, $\ldots$, $\mathrm{Enc}(V_B[N\text{-}1])$ to Alice.

Step 2: Alice computes $c_i=\mathrm{Enc}(r_i(V_B[i]-V_A[i])+i)$ for every i, then sends $\{(i, c_i)\}_i$ to Bob. However, r_i denotes a random number generated anew for every i.

Step 3: Bob decrypts the sent ciphertext, and, if the element is included in S_B, then outputs it as an element of the intersection.

In the theory of communication complexity it is known that if the size of the universal set is N then the cost of communication is given by $\Omega(N)$. If this is specialized to the case where n, the size of the set, satisfies $n \log N < N$, then an effective protocol can be configured as follows.

Similar to the above, consider the non-obfuscated method first.

Method 2

Input: Alice's input is S_A, Bob's input is S_B.

Step 1: Alice sends each element of S_A to Bob.

Step 2: Bob outputs the intersection of S_A and S_B.

The communication traffic volume in this method is given by $n \log N$. Therefore, if $N > n \log N$ then this method is more efficient than the method 1 in terms of both time complexity and communication complexity.

The obfuscated version of this method can be described as follows.

Secure set-intersection protocol (method 2)

Input: Alice's input is $S_A=\{a_1, \ldots, a_n\}$, pk,
Bob's input is $S_B=\{b_1, \ldots, b_n\}$, pk, sk.

Step 1: Bob sends $\mathrm{Enc}(b_l)$, $\mathrm{Enc}(b_2)$, $\ldots$, $\mathrm{Enc}(b_n)$ to Alice.

Step 2: Alice sends $\mathrm{Enc}(r_{ij}(b_i-a_j)+a_j)$ for every i, j. Here r_{ij} denotes a random number.

Step 3: Bob decrypts the sent ciphertext, and, if the plain text is included in S_B, then outputs it as an element of the intersection.

The communication complexity in this method is given by $\mathrm{O}(n^2)$, which is not necessarily efficient. The solution for this problem has been proposed by Freedman et al.[1]. By that, they have succeeded in reducing the communication traffic volume to $\mathrm{O}(n)$ by using the polynomial expressions of a set.

Applying the **bucket allocation** technique to this method can significantly improve time complexity.

3 Proposed method 1

In the method of Freedman et al., time complexity is not linear for n, and it is not satisfactory for use in the real world. Alternatively, Nojima and Kadobayashi[3] have proposed a method where the complexity is given by $\mathrm{O}(n)$.

3.1 Blind signature

Blind signature is a cryptographic protocol between two parties (signer and applicant). The signer possesses a signature key sk and a validation key vk, and the applicant possesses a message M and vk. This protocol enables the applicant to obtain the electronic signature $\mathrm{Sig}(sk, M)$ (may be abbreviated as $\mathrm{Sig}(M)$) without disclosing each other's information, sk or M (Figs. 3 and 4).

3.2 Method

Secure set-intersection protocol (Nojima-Kadobayashi)

Input: Alice's input is $S_A=\{a_1, \ldots, a_n\}$, pk, vk,
Bob's input is $S_B=\{b_1, \ldots, b_n\}$, vk.

Step 1: Alice sends H(Sig(a_1)), H(Sig(a_2)), …, H(Sig(a_n)) to Bob. Here H denotes a hash function.

Step 2: Bob and Alice operate blind signatures. Here Alice's input is *sk*, Bob' input is b_1, …, b_n. This protocol enables Bob to obtain H(Sig(b_1)), H(Sig(b_2)), …, H(Sig(b_n)).

Step 3: Bob obtains the intersection by comparing H(Sig(a_I)), H(Sig(a_2)), …, H(Sig(a_n)) and H(Sig(b_1)), H(Sig(b_2)), …, H(Sig(b_n)) (Fig. 5).

Chaum's blind signature scheme[4] can be used for blind signatures. In this method the complexity is given by O(n), which is very efficient.

3.3 Adaptability to IP-traceback

The previously introduced cryptographic protocols require computing modular exponentiation for the number of times that is proportional to n, the size of the set. However, in the IP-traceback scheme n means the number of packets, and it is impossible to compute modu-

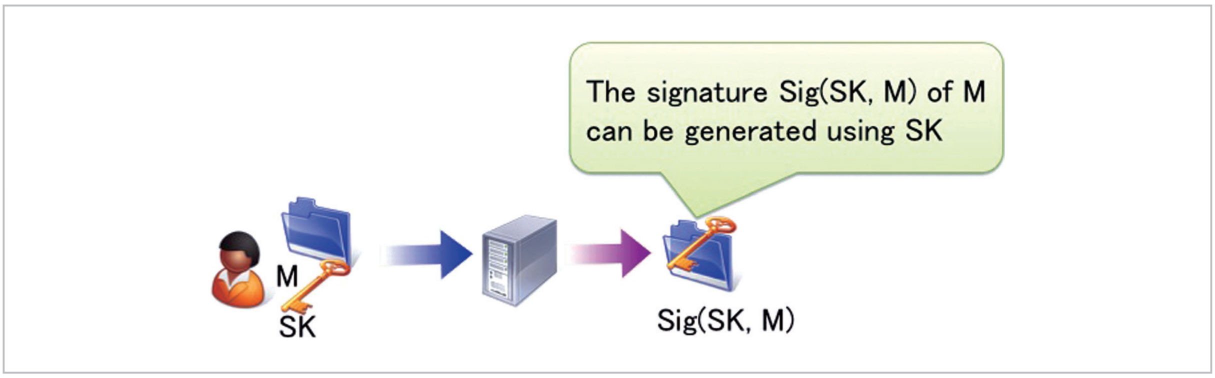

Fig.3 Description of electronic signature

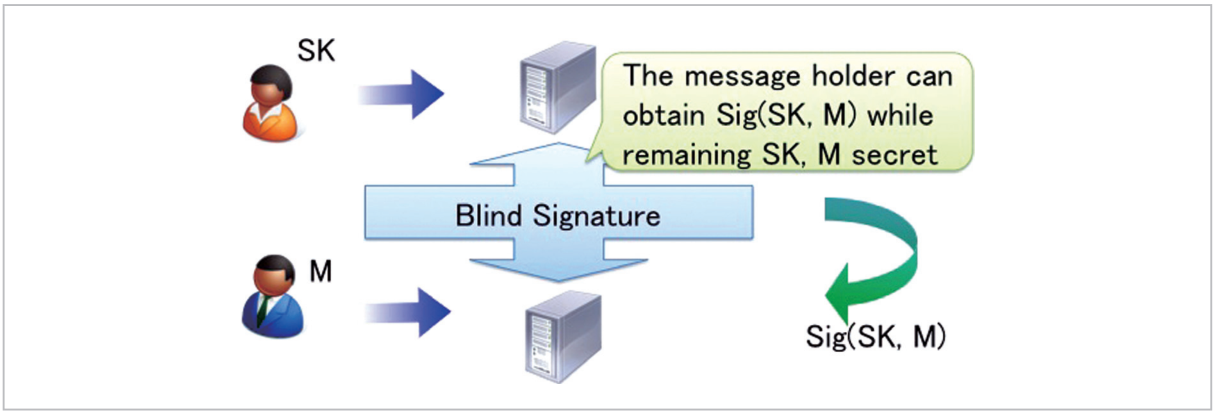
SK
The message holder can obtain Sig(SK, M) while remaining SK, M secret
Blind Signature
M
Sig(SK, M)

Fig.4 Description of blind signature

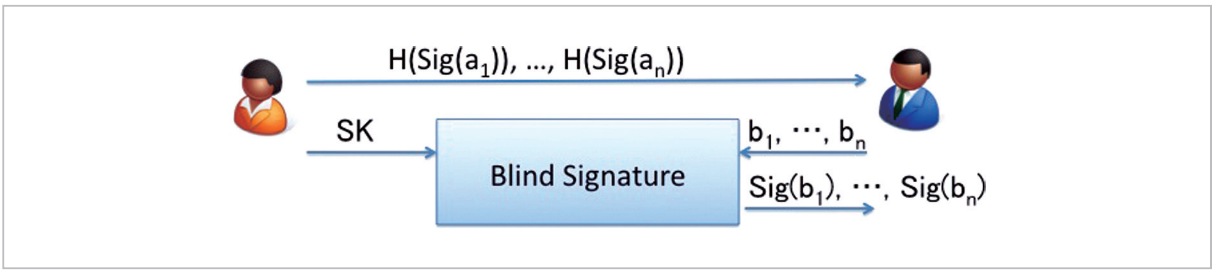

Fig.5 Secure set-intersection protocol based on blind signature

lar exponentiation for that number of times in reality.

For this reason, we have improved the method based on blind signatures, and propose a more realistic method in **4**.

4 Proposed method 2

4.1 Secret key encryption

Secret key encryption enables message M to be encrypted using a secret key sk. The ciphertext is given by $\mathrm{Enc}(sk, M)$. Here only the owner of the secret key sk can extract the message M from $\mathrm{Enc}(sk, M)$. On the contrary, a party that does not own sk cannot obtain any information related to M (Fig. 6). The typical secret key encryption schemes include DES and AES.

4.2 Oblivious symmetric encryption

Oblivious symmetric encryption protocol (hereafter OEP) is a cryptographic protocol between two parties (Alice and Bob).

Alice possesses the secret key sk that is used for the secret key encryption, and Bob possesses the message M.

This protocol enables to compute the ciphertext $C = \mathrm{Enc}(sk, M)$ without disclosing either party's information, sk or M, to the other party. Here, it is of course Bob who can obtain C, and Alice cannot obtain any information related to C at all (Fig. 7). We have successfully designed and developed OEP for the secret key encryption DES. The detail of this method is described later in this paper.

4.3 Application to IP-traceback

In privacy-preserving IP-traceback technology, Alice and Bob need to verify whether a is included in $A = \{a_1, \ldots, a_n\}$ without disclosing each other's information. This problem can be easily solved by OEP (Fig. 8).

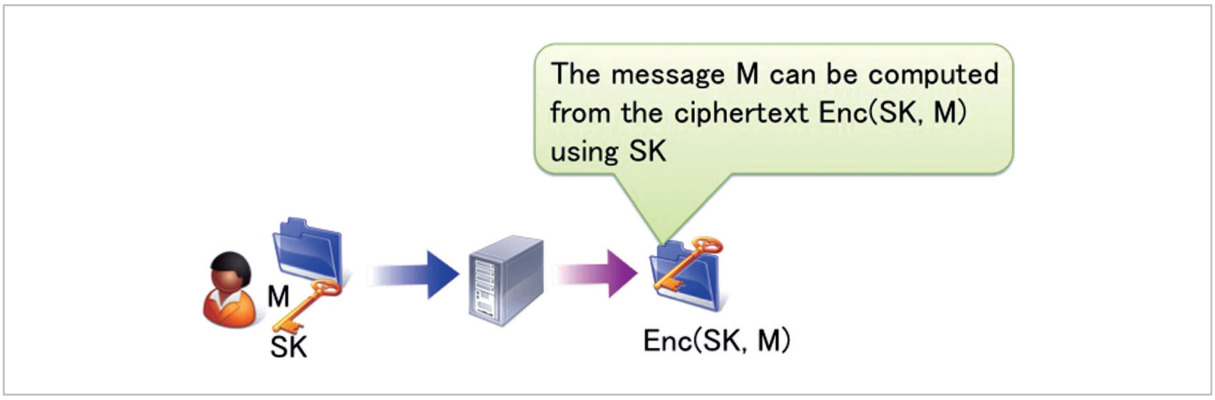
The message M can be computed from the ciphertext Enc(SK, M) using SK
M
SK
Enc(SK, M)

Fig.6 *Description of secret key encryption*

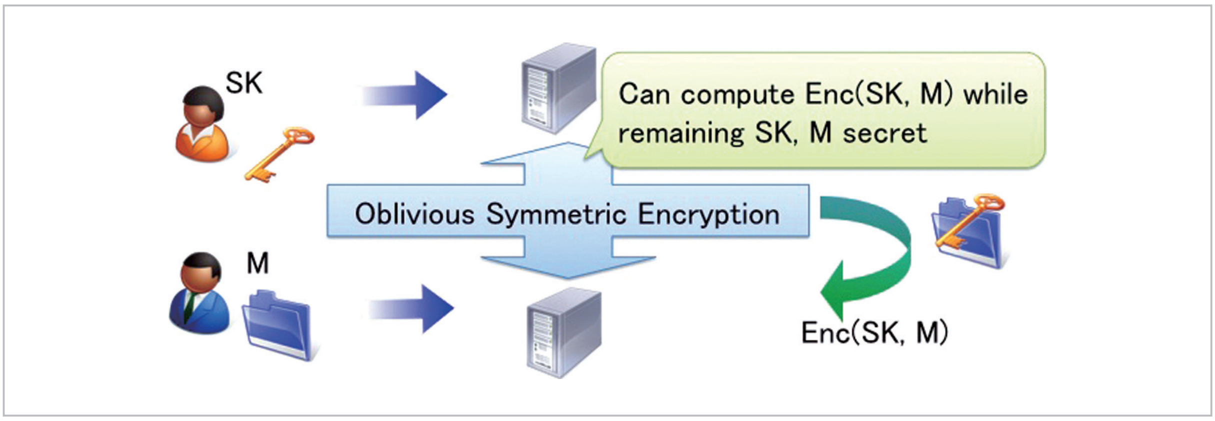
SK
Can compute Enc(SK, M) while remaining SK, M secret
Oblivious Symmetric Encryption
M
Enc(SK, M)

Fig.7 *Description of oblivious symmetric encryption protocol*

(1) Alice chooses the secret key sk used for the secret key encryption, and sends $\mathrm{Enc}(sk, a_1)$, …, $\mathrm{Enc}(sk, a_n)$ to Bob.
(2) Bob uses OEP to obtain $\mathrm{Enc}(sk, a)$. Then, if he finds an element that is equal to $\mathrm{Enc}(sk, a)$ in $\mathrm{Enc}(sk, a_1)$, …, $\mathrm{Enc}(sk, a_n)$, he determines that a is included in A.

As the use of OEP enables each party to hide sk or a from each other, Bob's secret information a will not be disclosed to Alice. In addition, as sk will not be disclosed to Bob, Alice's secret information A will also not be disclosed from n ciphertexts.

The role of OEP in this protocol is the same as the blind signatures in the proposed method 1. The advantage of this protocol is that Alice does not need to compute modular exponentiation when computing ciphertext of a_1, …, a_n. Therefore, it is suitable for the case where the number of packet n is vast, such as IP-traceback.

4.4 Extension

IP-traceback often uses bloom filter as a method to store the hash values of packets. This section introduces a method to apply OEP to the bloom filter based method.

Obfuscation of bloom filter based IP-traceback[5]

In privacy-preserving IP-traceback technology, Alice and Bob need to verify whether a is included in $A=\{a_1, …, a_n\}$ without disclosing each other's information. In the method introduced here, Alice's set is stored in a bloom filter. Let the length of array used for bloom filter be 2^m, and define the hash function to be used as $\mathrm{H}(x)=\mathrm{G}(\mathrm{Enc}(sk, x))$. G denotes a pseudo random number generator, and $\mathrm{H}_1(x)$ denotes the first m bit of $\mathrm{H}(x)$, …, $\mathrm{H}_k(x)$ denotes the last m bit of $\mathrm{H}(x)$. Therefore, the number of hash functions to be used is k.

(1) Alice chooses the secret key sk for the secret key encryption, and stores a_1, …, a_n by using $\mathrm{H}(x)=\mathrm{G}(\mathrm{Enc}(sk, x))$. Then she sends the encrypted bloom filter to Bob.
(2) Bob obtains the hash value $\mathrm{G}(\mathrm{Enc}(sk, a))$ by using OEP. Then, he verifies whether $a \in \{a_1, …, a_n\}$ is true by comparing with the encrypted bloom filter.

The security is maintained for the same reason as in the method that is not based on bloom filter. In other words, the use of OEP prevents the disclosure of sk and a, and as a result, Bob's secret information a will be prevented from being disclosed to Alice. On the contrary, as sk will not be disclosed to Bob, Alice's secret information A will not be disclosed from the bloom filter.

5 Practical configuration method[6]

5.1 Preparation: modified ElGamal

In order to configure DES based oblivious symmetric encryption (Oblivious-DES), consider a cryptographic scheme (Modified ElGamal) that is based on DDH assumption.

Let $G=<g>=<h>$ denote a group of order, where q is a prime number. DDH assumption is valid in this group. Define the secret key and public key as follows, respectively:

$$sk=x$$
$$pk=(q, g, h, g^x)$$

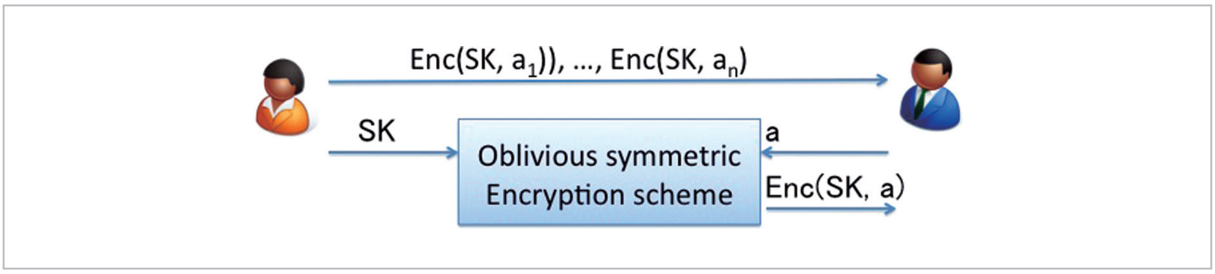
Enc(SK, a_1)), …, Enc(SK, a_n)
SK
Oblivious symmetric Encryption scheme
a
Enc(SK, a)

Fig.8 *Solution with the use of oblivious symmetric encryption protocol*

Here x denotes an element randomly chosen from Z_q. Define the encryption algorithm and decryption algorithm as follows:

$$\mathrm{Enc}(m; r) = (g^r, g^{rx}h^m)$$
$$\mathrm{Dec}(c_1, c_2) = c_2 / c_1{}^x$$

Let r denote an element randomly chosen from Z_q. Therefore,

$$\mathrm{Dec}(\mathrm{Enc}(m; r)) = h^m$$

Although the message space (logarithmic space) of this method is small, it is large enough to design oblivious-DES. In fact, the required message space, which is very small, is as follows:

$$\{0, 1, \ldots, 15\}$$

It is recommended to implement this method in such a way where the decryption algorithm could prepare the list $(h^0, h^1, \ldots, h^{15})$ in advance.

5.2 Proposed method

In order to design Oblivious-DES, it is necessary to design a Private-indexing protocol first. The concept of a Private-indexing protocol was invented by Naor and Nissim[7]. Their design uses Oblivious Transfer as a black box. On the other hand, we use a Modified ElGamal for the practical configuration. The advantage of using this method is great, as efficiency will be significantly improved. That is, when using Oblivious Transfer, generic ZK is required in order to maintain the security; however, when using Modified ElGamal, only the more efficient Σ-protocol is required for configuration.

5.2.1 Private-indexing protocol based on modified ElGamal

d sequences of ciphertext

$$\mathrm{Enc}(m_1), \ldots, \mathrm{Enc}(m_d)$$

are given by

$$\mathrm{Enc}_d(m_1, \ldots, m_d)$$

Let DB be an array with index from 0 to $2^{d_1} - 1$. The i^{th} value of DB is denoted as DB $[i]$, where DB $[i] \in \{0, 1\}^{d_2}$ for every i.

Suppose π_1, π_2 as a bit string with length d_1, and π_3, π_4 as a bit string with length d_2, then our aiming protocol is given by the function

$$f_{\mathrm{DB}}(\pi_1, \pi_2) = (\pi_3, \pi_4)$$

Here

$$\pi_3 + \pi_4 = \mathrm{DB}[\pi_1 + \pi_2]$$

is satisfied. Therefore,

As for the input π_1, the sender S obtains π_3
As for the input π_2, the recipient R obtains π_4

Suppose R possesses a public/secret key pair (pk, sk), and S possesses a public key pk. Under this presumption, our Private-indexing protocol is denoted as follows.

1. R sends $\mathrm{Enc}_{d_1}(\pi_2)$ to S. Here, the sequence of ciphertext consists of ciphertext containing 0s or 1s.
2. S computes $\mathrm{Enc}([\pi_1 + \pi_2])$ from $\mathrm{Enc}_{d_1}([\pi_1 + \pi_2])$.
   Then, S randomly chooses a bit string π_3 with length d_2, and for every $0 \leqq i \leqq 2^{d_1} - 1$ sends
   $$C^{(i)} = \mathrm{Enc}(\mathrm{r}_i([\pi_1 + \pi_2 - [i]]) + [\mathrm{DB}[i] + \pi_3])$$
   Here r_i is chosen randomly.
3. R decrypts the ciphertext to obtain
   $$\pi_4 = \mathrm{DB}[\pi_1 + \pi_2] + \pi_3$$

5.2.2 The basic configuring method of Oblivious DES

Consider two parties, S and R. S possesses a secret key of DES, k, and R possesses a 64bit plain text m. The goal of this protocol is to let R obtain the ciphertext DES (k, m). The Private-indexing protocol configures the Obvious DES protocol as below.

Initial phase

R and S operate as follows.

- As for the input m, R creates (sk, pk) and sends pk. Next, compute
  $$m' = (m_1', \ldots, m_{64}') = \mathrm{IP}(m)$$
  Then, let
  $$\mathrm{R_L}^{(0)} = (m_1', \ldots, m_{32}'),$$
  $$\mathrm{R_R}^{(0)} = (m_{33}', \ldots, m_{64}')$$

As for the input k, S computes the sub key $k^{(1)}, \ldots, k^{(16)}$.
Here, let

$$S_L^{(0)} = (0, \ldots, 0),$$
$$S_R^{(0)} = (0, \ldots, 0)$$

***i* th round (1 ≦ *i* ≦ 16)**

- S computes

$$E(S_R^{(i-1)}) + k^{(i)} = (\alpha_1, \ldots, \alpha_{48}) = (\beta_1, \ldots, \beta_8)$$

and R computes

$$E(R_R^{(i-1)}) = (\alpha_1', \ldots, \alpha_{48}') = (\beta_1', \ldots, \beta_8')$$

Here, let α_i, $\alpha_i' \in \{0, 1\}$, $\beta_1 = (\alpha_1, \ldots, \alpha_8)$, $\ldots$, $\beta_8 = (\alpha_{43}, \ldots, \alpha_{48})$, $\beta_1' = (\alpha_1', \ldots, \alpha_8')$, $\ldots$, $\beta_8' = (\alpha_{43}', \ldots, \alpha_{48}')$.

- S and R operate Private-indexing in parallel, which is described below:

$$f_{S_1}(\beta_1, \beta_1') = (\gamma_1, \delta_1)$$
$$\cdots$$
$$f_{S_8}(\beta_8, \beta_8') = (\gamma_8, \delta_8)$$

Here $S_1, \ldots, S_8$ denote S-box. Consider the following variables.

$$\varepsilon = (\gamma_1, \ldots, \gamma_8)$$
$$\zeta = (\delta_1, \ldots, \delta_8)$$

- S is replaced by

$$S_R^{(i)} = P(\varepsilon) + S_L^{(i-1)}$$
$$S_L^{(i)} = S_R^{(i-1)}$$

R is replaced by

$$R_R^{(i)} = P(\zeta) \oplus R_L^{(i-1)}$$
$$R_L^{(i)} = R_R^{(i-1)}$$

Final phase

- S sends

$$\eta = FP((S_L^{(16)}, S_R^{(16)}))$$

- R receives η and outputs

$$DES_k(m) = FP((R_L^{(16)}, R_R^{(16)})) \oplus \eta$$

6 Summary

This paper introduced privacy-preserving IP-traceback. Specifically, it proposed the configuration of DES based oblivious symmetric encryption, and presented its effectiveness for privacy-preserving IP-traceback. In fact, we have succeeded in implementing DES based oblivious symmetric encryption. Our future research projects include the design/implementation of efficient AES based oblivious symmetric encryption.

References

1 Michael J. Freedman, Kobbi Nissim, and Benny Pinkas, “Efficient Private Matching and Set Intersection,” EUROCRYPT 2004, 1–19.

2 Pascal Paillier, “Public-Key Cryptosystems Based on Composite Degree Residuosity Classes,” EUROCRYPT 1999, 223–238.

3 Ryo Nojima and Youki Kadobayashi, “On the Construction of the Set-Intersection Protocol from Blind Signatures,” ISEC 2008–9, 57–60.

4 David Chaum, “Blind Signatures for Untraceable Payments,” CRYPTO 1982, 199–203.

5 Ryo Nojima and Youki Kadobayashi, “Cryptographically Secure Bloom-Filters,” Transactions on Data Privacy 2(2), 131–139, 2009.

6 Ryo Nojima and Youki Kadobayashi, “Oblivious Symmetric Key Encryption and Its Application to IP-Traceback,” ISEC 2010-103, 199–203.

7 Moni Naor and Kobbi Nissim, “Communication preserving protocols for secure function evaluation,” STOC 2001, 590–599.

(Accepted June 15, 2011)

NOJIMA Ryo, *Dr. Eng.*

Senior Researcher, Security Fundamentals Laboratory, Network Security Research Institute

Cryptography