Improving Fairness and Privacy of Zhou-Gollmann's Fair Non-repudiation Protocol

Kwangjo Kim
ICU
58-4 Hwaam-dong
Yusong-ku, Taejon
305-348, Korea
kkj@icu.ac.kr

Sangjoon Park
ETRI
161 Kajong-dong
Yusong-ku, Taejon
305-600, Korea
sjpark@dingo.etri.re.kr

Joonsang Baek
ICU
58-4 Hwaam-dong
Yusong-ku, Taejon
305-348, Korea
mohi@icu.ac.kr

Abstract

We deal with two claws of Zhou-Gollmann's fair non-repudiation protocol. Firstly, their protocol divides a message into 2 parts, i.e., a key K and a ciphertext C. Then, C is delivered to the recipient, while K is submitted to TTP (Trusted Third Party). If the originator doesn't submit K to TTP, then the protocol appears to have no dispute between the originator and the recipient. However, the protocol depends on his action on whether the originator really submits K to TTP or not. We show that the originator can make the protocol unfair by using his advantageous position, and present how to improve the fairness of the protocol. Secondly, the protocol doesn't provide the message privacy. This means that additional protocols are required to transfer an important message in private. We propose an improved version of the protocol to guarantee the message privacy.

1. Introduction

When unforgeable evidence that a specific action occurred is required, non-repudiation service should be employed. This happens where sensitive paper documents such as contracts, bids, orders and cheques are stored, processed, and distributed in a digital form for EDI (Electronic Data Interchange), CALS (Commerce At the Light Speed) and EC (Electronic Commerce)systems. The goal of the non-repudiation service is *to collect, maintain, make available and validate irrefutable evidence concerning a claimed event or action in order to resolve dispute about the occurrence or non occurrence of the event or action* [9]. Non-repudiation service is composed of four distinct phases: evidence generation, evidence transfer, evidence verification and dispute resolution.

In the earlier non-repudiation protocol, it was focused what evidence both entities can have after a message originator and a recipient followed the predetermined protocol. The important goal of the latest non-repudiation protocol is not to make an entity in an advantageous position compared to other entity, *i.e, fairness* even if the protocol suddenly breaks in the middle. When sending a document, for example, an originator wants to receive a receipt from a recipient along with its document, while a recipient doesn't want to provide receipt before receiving all the document.

The originator sends all messages to the recipient only believing the honesty of the recipient while the recipient doesn't send a receipt to the originator. In this case, the recipient can deny the fact that he receives all the messages. The originator can't prove the fact that he sent all the messages even if he really sent them to the recipient. The non-repudiation protocol must provide a fair service in every step to both entities. To provide a fair service to the originator and the recipient, an exchange of information must be done simultaneously, *i.e.,* the information of originator's sending a message to the recipient and the information of the recipient's sending a receipt to the originator must be exchanged at the (almost) same time. One method to exchange information simultaneously is to use complicated cryptographic protocols[1][2][4][5][7][10], but they seem to be impractical.

The other method[12] is considered to utilize the reliable TTP (Trusted Third Party). This method is one of practical solutions, but the problem is high dependency on TTP in executing each step of the protocol compared to the previous method.

In 1996, Zhou and Gollmann proposed two fair non-repudiation protocols. One is to minimize the role of TTP[12] and the other is to use TTP as Delivery Agency[13]. In this paper, we analyze the former which we simply call it ZG's fair protocol. In the non-repudiation protocol based on TTP, it relays information between the originator and the recipient and issues certificate to the originator (recipient) at each step of the protocol. TTP plays an important role for the non-repudiation service. But, even if using TTP, ZG's fair protocol is designed to reduce the role of TTP as small as possible. It divides a message into 2 parts, *i.e.*, a key K and a ciphertext C. Then, C is delivered to the recipient and the recipient sends the receipt to the originator. To decrypt C, the recipient has to wait until the originator submits K to TTP. The originator, who is positioned in such an advantageous way, can make the protocol unfair. We deal with this unfairness at first. On the other hand, we consider that secrecy of messages for non-repudiation service is required. In ZG's fair protocol, anyone who can access TTP's public directory can get a key submitted to TTP and decrypt a message at his will. Hence, ZG's fair protocol doesn't provide message privacy. This means that additional protocols are required to send a secret message. We will suggest an improved version of ZG's fair protocol to support message privacy.

This paper consists of 5 Sections. In Section 2, we introduce ZG's fair protocol in brief. In Section 3, we point out the unfairness of ZG's fair protocol and suggest how to improve it by adding extra time limit in ZG's fair protocol. In Section 4, we propose enhancement of security service of ZG's fair protocol by adding public key distribution scheme such as Diffie-Hellman's one[6]. Finally, the concluding remarks are stated in Section 5.

2. Zhou-Gollmann's Protocol

The followings are the notation that will be used throughout this paper.

- $X||Y$: concatenation of two messages X and Y.
- $E(X, K, e)$: encryption of message X with K. ($E(\cdot)$ denotes a symmetric cryptosystem)
- $E(X, K, d)$: decryption of message X with key K.
- s_A : secret key of entity A for generating signature.
- p_A : public key of entity A for verifying signature.
- $S(X, s_A)$: digital signature of message X using s_A by entity A.
- A : originator of the non-repudiation exchange.
- B : recipient of the non-repudiation exchange.
- TTP : on-line TTP providing network services accessible to the public.
- M : message sent from A to B.
- C : ciphertext for message M, e.g., M encrypted under a key. ($C = E(M, K, e)$)
- K : message key defined by A.
- L : unique label which links to all messages of a particular protocol.
- f_{NRO} : flag information indicating NRO(Non-repudiation of Origin).
- f_{NRR} : flag information indicating NRR(Non-repudiation of Receipt).
- f_{SUB} : flag information indicating submission of a key.
- f_{CON} : flag information indicating confirmation of a key issued by TTP.
- $NRO = S(f_{NRO}||B||L||C, s_A)$: information of NRO.
- $NRR = S(f_{NRR}||A||L||C, s_B)$: information of NRR.
- $sub\_K = S(f_{SUB}||B||L||K, s_A)$: proof of submission of K by A.
- $con\_K = S(f_{CON}||A||B||L||K, s_T)$: confirmation of K issued by TTP.

2.1. ZG's fair protocol

A message is processed by splitting into its encryption key, K and its ciphertext, C. At first, an originator sends a ciphertext C to a recipient, and the recipient sends an acknowlegdement of the receipt(NRR) to the originator. Next, the originator submits his key K to TTP. It publishes K and its certificate con_K in TTP's public directory. The recipient gets K from TTP's public directory and decrypts the ciphertext C by the key K, and the originator also gets K from the public directory and stores it with the receipt. In each step, all messages are connected by the link label.

A, B and TTP have their own private keys, s_A, s_B and s_T for generating signatures and their relevant public keys, p_A, p_B and p_T for verifying signatures, respectively.

The protocol is described in each step as follows :

1. $A \rightarrow B$:	f_{NRO}, B, L, C, NRO
2. $B \rightarrow A$:	f_{NRR}, A, L, NRR
3. $A \rightarrow TTP$:	f_{SUB}, B, L, K, sub_K
4. $B \longleftrightarrow TTP$:	$f_{CON}, A, B, L, K, con\_K$
5. $A \longleftrightarrow TTP$:	$f_{CON}, A, B, L, K, con\_K$

The role of TTP is not an Delivery Agent but a Certification Agency that issues a certification of registry for A's submitted key K.

2.2. Dispute Resolution

Disputes can arise over the origin and receipt of a message, M. The first case occurs that A claims to deny sending M to B. The second case occurs that B claims to deny receiving M from A.

Non-repudiation of Origin

If A claims that he did not send M, B submits M, C, K, L and the non-repudiation evidence NRO, con_K to a judge. The judge can verify that M was sent by A by the following process :

- checks that con_K is TTP's signature on $f_{CON}||A||B||L||K$.
- checks that NRO is A's signature on $f_{NRO}||B||L||C$.
- checks that $M = E(C, K, d)$.

Non-repudiation of Receipt

If B claims that he hasn't received M from A, A submits M, C, K, L and the non-repudiation evidence NRR, con_K to the judge. The judge can verify that B has received M by the following process :

- checks that con_K is TTP's signature on $f_{CON}||A||B||L||K$.
- checks that NRR is B's signature on $f_{CON}||A||L||C$.
- checks that $M = E(C, K, d)$.

2.3. Time Limit

First we consider that even if A has received f_{NRR}, A, L and NRR from B, he wouldn't submit K and sub_K to TTP in ZG's fair protocol. In this case, A's NRR will become meaningless because he did not receive con_K from TTP. B must keep f_{NRO}, B, L, C and NRO which A has sent before. We consider that B deletes all informations which he received from A. Later, A submits f_{SUB}, B, L, K and sub_K to TTP and TTP opens f_{CON}, A, B, L, K and con_K in the public directory. Thus, A can get confirmation certificate, con_K indicating that B has received M. Since B deletes the ciphertext C, B does not get the plaintext M. Moreover, TTP has to store K and sub_K forever. This makes the protocol hard to implement. To solve this, we set a deadline T to limit the time con_K and K can be accessed by the public. The protocol is extended as follows :

$$\begin{aligned} NRO &= S(f_{NRO}||B||L||T||C, s_A) \\ NRR &= S(f_{NRR}||A||L||T||C, s_B) \\ sub\_K &= S(f_{SUB}||B||L||T||K, s_A) \\ con\_K &= S(f_{CON}||A||B||L||T||T_0||K, s_T) \end{aligned}$$

1. $A \rightarrow B$:	f_{NRO}, B, L, T, C, NRO
2. $B \rightarrow A$:	f_{NRR}, A, L, NRR
3. $A \rightarrow TTP$:	$f_{SUB}, B, L, T, K, sub\_K$
4. $B \longleftrightarrow TTP$:	$f_{CON}, A, B, L, T_0, K, con\_K$
5. $A \longleftrightarrow TTP$:	$f_{CON}, A, B, L, T_0, K, con\_K$

T_0 in con_K is the time stamp to indicate when the confirmed key has actually been made available to the public. If B does not agree with the deadline T, the protocol stops at Step 2. If sub_K and K don't reach TTP by the deadline T, then B deletes NRO and C in his storage.

3. Unfairness of ZG's fair protocol

In this section we will show that ZG's fair protocol with the time limit T may cause another problem. We consider that A sends K just before the time T. con_K may be deleted just after the time it is registered in the public directory. So, B keeps on monitoring the TTP's public directory around time T. At this time, A may disturb the network or computer system to prevent B from receiving con_K from the directory. This may arise since only the originator has capability to register K in TTP's directory and the recipient is in disadvantageous position than the originator. Now, we consider that B sets the valid time limit, $T_1(< T)$ of NRR in addition to public accessible time limit T.

Then, ZG's fair protocol is modified as follows :

$$\begin{aligned} NRO &= S(f_{NRO}||B||L||T||C, s_A) \\ NRR &= S(f_{NRR}||A||L||T||T_1||C, s_B) \\ sub\_K &= S(f_{SUB}||B||L||T||K, s_A) \\ con\_K &= S(f_{CON}||A||B||L||T||T_0||K, s_T) \end{aligned}$$

1. $A \rightarrow B$:	f_{NRO}, B, L, T, C, NRO
2. $B \rightarrow A$:	f_{NRR}, A, L, T_1, NRR
3. $A \rightarrow TTP$:	$f_{SUB}, B, L, T, K, sub\_K$
4. $B \longleftrightarrow TTP$:	$f_{CON}, A, B, L, T_0, K, con\_K$
5. $A \longleftrightarrow TTP$:	$f_{CON}, A, B, L, T_0, K, con\_K$

If there is enough time interval between T_1 and T, B can receive con_K from TTP's public directory at his convenient time between T_1 and T. If con_K is not posted before T_1 in TTP's public directory, A's NRR is no more valid. So, B can delete NRO.

The Non-repudiation of Origin in this modified protocol is the same as that of ZG's fair protocol. The Non-repudiation of Receipt is modified as follows :

Non-repudiation of Receipt

A submits M, C, K, T, T_0, T_1, L and the evidence NRR, con_K to the judge. The judge can confirm that B has received M by the following steps :

(1) checks that con_K is the TTP's signature on $f_{CON}||A||B||L||T||T_0||K$.

(2) checks that NRR is B's signature on $f_{CON}||A||L||T||T_1||C$.

(3) checks that $T_0 < T_1 < T$.

(4) checks that $M = E(C, K, d)$.

A must register sub_K to TTP before the time T_1 which was set by B in NRR. If, after the time T_1 elapses, A sends sub_K to TTP, TTP rejects the originator's claim that con_K is published in the public directory of TTP. If con_K will not be registered within T_1 in the public directory, he can delete NRO in his memory. This approach is highly dependent on the time information. The sharing time between A, B and TTP is very important.

A and B may have independent time. It will be difficult to synchronize two clocks. It needs setting up a clock manager to synchronize between two entities. In practice, we can get global clock information through a satellite such as GPS (Global Positioning System).

4. Adding Message Privacy

In ZG's fair protocol, anyone can decrypt C transferring from A to B since the corresponding key K is posted in TTP's public directory. If A wants to send M to B privately, it requires additional protocols. In general the level of security of a message in non-repudiation service is higher than that of a normal message. It is desirable to provide privacy and non-repudiation services together.

In this Section, we describe how to deliver C in private and to distribute K in safe by introducing Diffie-Hellman public key distribution scheme. In our combined protocol, each entity is assumed to have a signature key in ZG's fair protocol and a DH encryption key for key distribution. We need to maintain two cipher systems : one for signature scheme and the other for key distribution. Due to the corresponding system complexity, the generalized ElGamal-type signature is more efficient. Each entity can have only one key not only for generating a signature but also for encrypting session key K for message privacy.

Each entity must generate public and secret key pairs for message privacy. Let p and q be large primes where $p = 2q + 1$ and g be a primitive element over $GF(p)$. The secret and public keys of A and B are :

$$p_A = g^{s_A} \bmod p,\ p_B = g^{s_B} \bmod p.$$

By using the public key p_B, the key distribution process of an entity A for message privacy key K is stated as follows.

- Key generation by an entity A
  - generates a random number, r $(0 < r < p - 1)$.
  - computes $K = p_B^r \bmod p$.
  - computes $K_{sub} = g^r \bmod p$.
  - transmits K_{sub} to B.
- An entity B's computation for message privacy key, K : $K = K_{sub}^{s_B} = g^{rs_B} \bmod p$.

Only B who has a secret key s_B can compute message privacy key, K from K_{sub}. Thus, A can believe that only B can recover the encrypted message. But, B can't confirm that K_{sub} comes from A. In order to solve this, TTP confirms A's signature on K_{sub} and B receives TTP's signature on K_{sub}.

The non-repudiation protocol, which can provide message privacy using DH public key distribution scheme, is described as :

$$\begin{aligned} NRO &= S(f_{NRO}||B||L||T||C, s_A) \\ NRR &= S(f_{NRR}||A||L||T||T_1||C, s_B) \\ sub\_K &= S(f_{SUB}||B||L||T||K_{sub}, s_A) \\ con\_K &= S(f_{CON}||A||B||L||T||T_0||K_{sub}, s_T) \end{aligned}$$

1. $A \rightarrow B$:	f_{NRO}, B, L, T, C, NRO
2. $B \rightarrow A$:	f_{NRR}, A, L, T_1, NRR
3. $A \rightarrow TTP$:	$f_{SUB}, B, L, T, K_{sub}, sub\_K$
4. $B \longleftrightarrow TTP$:	$f_{CON}, A, B, L, T_0, K_{sub}, con\_K$
5. $A \longleftrightarrow TTP$:	$f_{CON}, A, B, L, T_0, K_{sub}, con\_K$.

In step 3, A sends K_{sub} and its signature sub_K to TTP. After checking that sub_K is A's signature on K_{sub}, TTP publishes K_{sub} and TTP's signature, con_K into the public directory.

In step 4, B gets con_K from the public directory and confirms that K_{sub} was sent by A. B computes message encryption key, $K = K_{sub}^{s_B} = p_B^r \bmod p$ by using his own secret key, s_B. In our proposed protocol, A must keep r secret for non-repudiation verification process such as $K_{sub} = g^r \bmod p$. Even TTP is not able to derive K from K_{sub} and thus not able to decipher the message. During the process, TTP is simplified to check A's submitted key, K_{sub} and L.

Non-repudiation of Origin

B submits M, C, K_{sub}, L, T, T_0 and the evidence, NRO, con_K to the judge. The judge confirms that M was sent by A as follows :

(1) checks that con_K is TTP's signature on $f_{CON}||A||B||L||T||T_0||K_{sub}$,

(2) confirms from B that the corresponding key of K_{sub} is K,

(3) checks that NRO is A's signature on $f_{NRO}||B||L||T||C$, and

(4) checks that $M = E(C, K, d)$.

In step (2), B does not reveal s_B and proves to the judge whether $K = K_{sub}^{s_B} \bmod p$. The proving method is the same as the confirmation protocol in Chaum's undeniable signature[3].

Non-repudiation of Receipt

A submits $M, C, K_{sub}, T, T_0, T_1, L, r$ and the non-repudiation evidence NRR, con_K to the judge.

The judge confirms that B must receive M by the following steps :

(1) checks that con_K is TTP's signature on $f_{CON}||A||B||L||T||T_0||K_{sub}$,

(2) checks that $K_{sub} = g^r \bmod p$ from r, K_{sub} and computes $K = p_B^r \bmod p$ by B's public key, p_B,

(3) checks that NRR is B's signature on $f_{CON}||A||L||T||T_1||C$,

(4) checks that $T_0 < T_1 < T$, and

(5) checks that $M = E(C, K, d)$.

5. Concluding Remarks

In this paper, we have dealt with two claws of ZG's fair protocol. To sum up, the first problem with ZG's fair protocol is the unfairness. The originator can make ZG's protocol to be unfair since the originator can be in a more advantageous position than the recipient. The second problem is the message privacy. The originator sent a ciphertext to the recipient and its corresponding key to TTP's public directory at a later time. Anyone who can access the public directory can get the key to decrypt the ciphertext. Our solution for the unfairness problem is to set up time limit (T_1) of the Non-repudiation receipt of message (NRR).

Also, by the introduction of Diffie-Hellman key distribution scheme in ZG's fair protocol, our protocol has made it impossible to recover encryption key from TTP's public directory. This leads to private message delivery. It can be seen that the degree of dependency on TTP and rate of communication overhead in our improved protocol are as small as those of ZG's fair protocol.

References

[1] M. Ben-Or, O. Goldreich, S. Micali, and R. Rivest, "A Fair Protocol for Signing Contracts", *IEEE Transactions on Information Theory*, 36(1):40-46, January 1990.

[2] E. F. Brickell, D. Chaum, I. B. Damgard, and J. van de Graaf, "Gradual and Verifiable Release of a Secret", *In Advances in Cryptology: Proceedings of Crypto'87, LNCS 293*, pp. 156-166, Springer-Verlag, 1988.

[3] D. Chaum, "Zero-Knowledge Undeniable Signatures", *In Advances in Cryptology: Proceedings of Eurocrypt'90, LNCS 473*, pp. 458-464, Springer-Verlag, 1991.

[4] R. Cleve, "Controlled Gradual Disclosure Schemes for Random Bits and Their Applications", *In Advances in Cryptology: Proceedings of Crypto'89, LNCS 435*, pp. 573-588, Springer-Verlag, 1990.

[5] I. B. Damgard, "Practical and Provably Secure Release of a Secret and Exchange of Signatures", *In Advances in Cryptology: Proceedings of Eurocrypt'93, LNCS 765*, pp. 200-217, Springer-Verlag, 1994.

[6] W. Diffie and M. Hellman, "New Directions in Cryptography", *IEEE Trans. on Information Theory*, Vol.22, pp. 644-654, 1976.

[7] S. Even, O. Goldreich, and A. Lempel, "A Randomized Protocol for Signing Contracts", *Communications of the ACM*, Vol. 28, No. 6, pp. 637-647, June 1985.

[8] L. Harn and Y. Xu, "Design of Generalized ElGamal Type Digital Signature Schemes Based on Discrete Logarithm", *Electronics Letters*, Vol.30, No.24, pp. 2025-2026, Nov 1994.

[9] ISO/IEC JTC1, Information Technology - Open Systems Interconnection - Security Frameworks in Open Systems, Part 4: Non-repudiation, ISO/IEC DIS 10181-4, April 1995.

[10] T. Okamoto and K. Ohta, "How to Simultaneously Exchange Secrets by General Assumptions", *In Proceedings of 2nd ACM Conference on Computer and Communications Security*, pp. 184-192, November, 1994.

[11] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystem", *Communications of the ACM*, Vol. 21, No. 2, pp. 120-126, 1978.

[12] J. Zhou and D. Gollmann, “A Fair Non-repudiation Protocol”, *In Proceedings of 1996 IEEE Symposium on Security and Privacy*, pp.55-61, May 6–8, 1996.

[13] J. Zhou and D. Gollmann, “Observations on Non-repudiation”, *In Advances in Cryptology: Proceedings of Asiacrypt’96, LNCS 1163*, pp. 133-144, Springer-Verlag, 1996.