An Efficient Certified Email Protocol*

Jun Shao[1], Min Feng[2,**], Bin Zhu[2], and Zhenfu Cao[1]

[1] Department of Computer Science and Engineering
Shanghai Jiao Tong University
200030, Shanghai, China P.R.C
[2] Microsoft Research Asia
100080, Beijing, China P.R.C
minfeng@microsoft.com

Abstract. A certified email protocol, also known as a non-repudiation protocol, allows a message to be exchanged for an acknowledgement of reception in a fair manner: a sender Alice sends a message to a receiver Bob if and only if Alice receives a receipt from Bob. In this paper, we present a novel approach to combine the authorized Diffie-Hellman key agreement protocol with a modified Schnorr signature effectively to construct our certified email protocol. Our proposed certified email protocol is an optimistic protocol, with an off-line trusted third party being involved only when a party cheats or the communication channel is interrupted during exchange. We also compare our protocol with other optimistic certified email protocols, and conclude that our certified email protocol is the most efficient optimistic certified email protocol.

Keywords: Certified Email Protocol, Fair Exchange Protocol, D Optimistic Fair Exchange Protocol.

1 Introduction

We all have the following experience: when a registered letter arrives, we receive the letter if and only if we have signed an acknowledgement of reception. The two actions, i.e., signing an acknowledgement and receiving the letter occur simultaneously. In an electronically connected world, emails are used widely. Most people prefer emails to snail mails in communicating with others due to convenience and fast delivery offered by the email. An email system should also provide the same function as the registered letter that a receiver has to sign an acknowledgement of reception before a registered email can be read. Unlike the case of registered letters, the two actions, i.e., signing and receiving, cannot occur simultaneously in an email system due to its distributed nature. The protocols used in an email system, or any protocols in general, are asynchronous by nature. How can we provide the "registered letter" service in a distributed environment? The answer is the certified email protocol, also known as the non-repudiation

* This work was done when Jun SHAO was an intern at Microsoft Research Asia.
** Corresponding author.

J. Garay et al. (Eds.): ISC 2007, LNCS 4779, pp. 145–157, 2007.

protocol[1]. A certified email protocol enables a fair exchange of a message and an undeniable receipt between two untrusted parties over a network such as the Internet.

In addition to certified emails, a certified email protocol can also be used in many other applications. One application is to secure the itinerary of a mobile agent [39], where a certified email protocol is applied between two adjacent hosts when a mobile agent passes from one host to the other. The non-deniable message and receipt offered by a certified email protocol are used to identify the origin of an attack if the itinerary of the mobile agent is altered. Another application is to encourage people to share or propagate contents such as self-created movies or advertisements with others, where a certified email protocol is used to assure that users who share contents with others would get awards by redeeming the receipts from content receivers.

Due to its usefulness, the certified email protocol has been studied widely by the cryptography research community. In fact, the problem addressed by the certified email protocol is essentially a subset of the problem addressed by the fair exchange protocol, where the exchanged items are not necessarily restricted to messages and receipts as in certified email protocols, yet other digital items can also be exchanged in the fair exchange protocol. For example, both parties can exchange signatures signed by each individual party in a fair exchange protocol. As a result, most of the techniques used in fair exchange protocols can also be used in certified email protocols.

Depending on the availability and setting of a Trusted Third Party (TTP), fair exchanges can be classified into the following four types: (1) without a TTP, (2) with an inline TTP, (3) with an online TTP, and (4) with an off-line TTP. For the first type of fair exchanges, Even and Yacobi [23] proved as early as in 1980 that it is impossible to realize fairness in a deterministic two-party fair exchange protocol. Existing protocols can provide only partial fairness: computational fairness [18,21,24] or probabilistic fairness [12,27]. These protocols are, however, too complex and inefficient to be applied in practical applications. For the second type, the TTP acts as an intermediary between the sender and the receiver, and the entire message is sent through the TTP [15,16]. An inline TTP can provide full fairness since all exchanged messages are fully controlled by the TTP. The TTP, however, may become a performance bottleneck, especially when many large messages have to forward at the same time. An online TTP is similar to an inline TTP, where the TTP must be available for the entire lifetime of the exchange. In such a setting, the TTP does not need to forward the entire message. Only the signaling information such as the cryptographic key is processed and forwarded by the TTP. For the last type, also known as the *optimistic* protocol, the TTP is involved only if one of the parties behaves maliciously or the communication channel is interrupted during execution of the exchange protocol.

[1] Some researchers [35,36] think that these two protocols are different. If we do not consider message's confidentiality, both protocols can be considered as the same since they both address the same problem: exchanging a message and a receipt in a fair manner.

This property is very desirable and practical in many applications, including the distributed environment mentioned above. The last type, therefore, has attracted more researchers' attention than the other three types. Up to now, many techniques have been proposed to address the fourth type of fair exchanges, such as the escrow and verifiable escrow scheme [5], the verifiable encryption [10, 1], the verifiable confirmation of signatures [17], the convertible signatures [11, 28], the designated verifier proofs [25], the cross validation [37, 38], the gradational signature [32], the sequential two-party multi-signature scheme [31], the verifiable and recoverable encrypted signature [29], and the verifiably committed signatures [20].

As mentioned before, most techniques used in a fair exchange protocol can also be used in a certified email protocol. There also exist many generic certified email protocols [34, 35, 36], where generic encryption and signature primitives are used. These generic certified email protocols usually utilize the following approach: first encrypting a message m by a symmetric encryption scheme, then encrypting the key used in the symmetric encryption by a public key encryption scheme with the TTP's public key, and finally signing the resulting ciphertext by a signature scheme with the sender Alice's private key. When the receiver Bob receives the signature, he first checks validity of the received signature. If it is valid, he sends a receipt to Alice to indicate that Bob has received the message. Bob's interest is protected since if Alice refuses to reveal the exchanged message m, the TTP Charlie can reveal the message m for him.

The most efficient existing certified email protocol with an off-line TTP is, to the best of our knowledge, the scheme proposed in [35, 36]. In this paper, we propose a novel and more efficient certified email protocol with an off-line TTP. Our scheme uses the technique of authorized key agreement. We believe that we are the first in applying this technique in a fair exchange protocol.

1.1 Our Contribution

The protocol to be proposed in the paper is a certified email protocol with an off-line TTP. The major contribution of this paper is that a novel approach is used to encrypt a message in a certified email protocol. Unlike other certified email protocols with an off-line TTP, which use the TTP's public key to encrypt a randomly selected message encryption key so that the TTP can extract the message encryption key to reveal the message in the execution of the dispute protocol, our protocol encrypts a message with a key shared between the sender and the TTP, yet without involving the TTP during the exchange. The step to apply a public key encryption scheme to encrypt the message encryption key used in other certified email protocols is removed in our protocol, resulting in a more efficient protocol.

The well-known authorized Diffie-Hellman key agreement [19] is used in our scheme to achieve the goal to share the message encryption key between the sender and the TTP. Like the protocols proposed in [34, 35, 36][2], our protocol is

[2] Two fair exchange protocols are proposed in [34]. One is with an online TTP, and the other with an off-line TTP. In this paper, we consider only the off-line protocol.

fair and optimistic. Compared with the existing certified email protocols [2, 4, 34, 32, 35, 36], our protocol enjoys the following properties:

Fairness: Like other certified email protocols, our protocol guarantees fairness, i.e., a malicious party cannot gain any advantage over the other party in exchange of a message and a receipt. Detailed security analysis and discussion are given in Section 3.

Optimism: Compared to the scheme proposed in [2], the TTP in our protocol is involved only when one party conducts malicious behaviors or the communication channel is interrupted during exchange. In other words, our protocol is an optimistic protocol.

TTP's Statelessness: The TTP does not need to store any state information in executing our protocol to deal with disputes between the two parties.

High Performance: Our protocol has the smallest computational and communicational cost among all certified email protocols. Comparison with typical existing certified email protocols is given in Section 4.

Note that we do not deal with the subtle issue of timely termination addressed by [5, 6]. We would like to point out that the techniques used in [5, 6] to deal with this subtle issue can be added easily to our protocol to resolve this problem. Furthermore, we assume that there exist reliable channels between the users and the TTP.

1.2 Organization

The rest of this paper is organized as follows. In Section 2 our novel certified email protocol is described in detail, followed by the discussion of security of our protocol in Section 3. Comparison of our protocol with the certified email protocols proposed in schemes in [35, 36] is given in Section 4. We conclude the paper in Section 5.

2 Our Proposed Protocol

This paper focuses on certified email protocols without considering confidentiality of the message m. Confidentiality can be achieved easily by applying an encryption scheme to the message m if needed. Before describing our protocol, we would like to describe a modified Schnorr signature scheme [7] which will be used in our certified email protocol.

2.1 Modified Schnorr Signature Scheme

The following signature scheme is based on Schnorr signature scheme [40] which is proved to be secure against the adaptively chosen message attack in the random oracle model [14] with the discrete logarithm assumption. It consists of the following three algorithms: `Setup`, `Sign`, and `Verify`.

Setup. It takes as input a security parameter 1^k and outputs a public key $(G, q, g, H(\cdot), y)$ and a secret key x, where q is a large prime, G is a finite

cyclic group with the generator g of order q, $H(\cdot)$ is a cryptographic hash function: $\{0,1\}^* \rightarrow Z_q^*$, and $y = g^x \in G$. $\mathcal{M}$ is the domain of messages.

Sign. To sign a message $m \in \mathcal{M}$, the following operations are applied: (1) choose a random $r \in Z_q^*$, (2) compute $R = g^r \in G$, and (3) set the signature to be $\sigma = (R, s)$, where $s = r + xH(m||R||y) \bmod q$.

Verify. To verify a signature σ for message m, the verifier checks $g^s \stackrel{?}{=} Ry^{H(m||R||y)} \in G$. If the equation holds, the signature is valid and outputs $b = 1$; otherwise, the signature is invalid and outputs $b = 0$.

2.2 Our Proposed Protocol

Our certified email protocol consists of the following three sub-protocols: the `setup sub-protocol`, the `exchange sub-protocol`, and the `dispute sub-protocol`. Assume that Alice is the sender, Bob is the receiver, and Charlie is the TTP. We also assume that the public key of the Certification Authority (CA) and the three parties are known to everyone. Let m denote the sent message and let σ_B denote the receipt.

Setup Sub-protocol. In our certified email protocol, first choose the system parameters (q, G, g), where q is a large prime, and G is a gap Diffie-Hellman (GDH) group[3] with the generator g whose order is q. Then Charlie select his random private key $x_c \in Z_q^*$, and computes and publishes the corresponding public key $y_c = g^{x_c} \in G$.

Alice also selects her random private key $x_a \in Z_q^*$, and computes the corresponding public key $y_a = g^{x_a} \in G$. But, she registers her public key and her system parameter with a CA to get her certificate C_A which binds her identity ID_A with the corresponding public key (q, G, g, y_a).

Exchange Sub-protocol. In this protocol, Alice sends to Bob a message m with the message description Dsc_m[4], and receives a receipt from Bob. The

[3] We call a finite cyclic group G, with the generator g whose order is prime q, is a gap Diffie-Hellman (GDH) group if the following first problem can be solved in polynomial time but no p.p.t. algorithm can solve the following second problem with non-negligible advantage over random guess within polynomial time [30].

Decisional Diffie-Hellman Problem. Given $(g, g^a, g^b, g^c) \in G * G * G * G$, decide whether $c = ab \in Z_q^*$, where a, b, c are three random numbers in Z_q^*. If $c = ab \in Z_q^*$, then (g, g^a, g^b, g^c) is a Decisional Diffie-Hellman (DDH) tuple.

Computational Diffie-Hellman Problem. Given $(g, g^a, g^b) \in G * G * G$, compute $g^{ab} \in G$, where a, b are two random numbers in Z_q^*.

[4] The description Dsc_m will enable a human being to verify a message. A simple description is the hash value of the message. The actual description depends on the application. When used in the application to encourage sharing multimedia, Dsc_m may be a description of the multimedia content such as its title, creator, etc. We note that knowledge of the description Dsc_m does not disclose its message m.

message description Dsc_m is used to check if a decrypted message matches its description. In the following description, $(\mathcal{E}_k(\cdot), \mathcal{D}_k(\cdot))$ is a pair of symmetric encryption and decryption operations with the encryption key k. $H(\cdot)$, $H_1(\cdot)$ and $H_2(\cdot)$ are cryptographic hash functions.

1. Alice first chooses a random number $r \in_R Z_q^*$, and computes

$$R_1 = g^r \in G, \quad R_2 = y_c^r \in G, \quad R' = H(R_2), \quad k = H_1(R_2),$$

$$C = \mathcal{E}_k(m), \quad s_A = r + x_a H_2(C||Dsc_m||ID_A||ID_B||ID_C||R_1||R'||y_a) \bmod q.$$

   Alice then sends $(C_A, R_1, R', C, Dsc_m, s_A)$ to Bob, where C_A is Alice's certificate obtained with the setup sub-protocol.
2. On receiving $(C_A, R_1, R', C, Dsc_m, s_A)$ from Alice, Bob first validates Alice's certificate C_A, and then checks if the following equation holds,

$$g^{s_A} \stackrel{?}{=} R_1 y_a^{H_2(C||Dsc_m||ID_A||ID_B||ID_C||R_1||R'||y_a)} \in G.$$

   If both checks are fine, Bob sends to Alice his signature σ_B on

$$(R_1, R', C, Dsc_m, s_A, ID_A, ID_B, ID_C).$$

3. Upon receiving σ_B from Bob, Alice first validates Bob's signature σ_B. If passes, Alice sends R_2 to Bob.
4. Upon receiving R_2, Bob computes the key $k = H_1(R_2)$ and uses it to decrypt the encrypted message C previously received to obtain the wanted message $m = \mathcal{D}_{H_1(R_2)}(C)$. The decrypted message m is considered as *valid* if and only if m *does* match the message description Dsc_m previously received. If he does not receive R_2, or $R' \neq H(R_2)$, or the decrypted message m does not match its description Dsc_m, Bob can invoke the dispute protocol.

Remark 2.1. R_1 will be used as a part of the key material in the Diffie-Hellman key agreement in the `dispute protocol` to be described later, and R_2 is the resulting key of the Diffie-Hellman key agreement. (R_1, s_A) is in fact a signature on $(C, R', Dsc_m, ID_A, ID_B, ID_C)$ corresponding to the public key y_a obtained by using the modified Schnorr signature scheme. Bob's signature σ_B in Step 2 above can be any type of signature.

Alice can use receipt σ_B she receives from Bob to prove to another person John that Bob has received the message m from her with the following procedure:

- Alice sends John $(\sigma_B, R_1, R', C, Dsc_m, s_A, C_A, ID_B, ID_C, m, R_2)$
- John checks whether
  - m is consistent with Dsc_m,
  - σ_B, s_A, C_A are valid,
  - $C = \mathcal{E}_{H_1(R_2)}(m)$,
  - $R' = H(R_2)$,
  - (g, R_1, y_c, R_2) is a Decisional Diffie-Hellman (DDH) tuple.

If all the above checks pass, John is convinced that Bob indeed receives the message m from Alice.

Our protocol uses a gap Diffie-Hellman group. Alice can determine whether (g, R_1, y_c, R_2) is a DDH tuple or not by using some special algorithms such as pairing. In some applications, Alice may need to prove *only* to the TTP that Bob has received message m, i.e., John is always the TTP. In this case, the protocol is the same as described above except that the gap Diffie-Hellman group can be replaced with a finite cyclic group whose CDH problem is computationally hard[5], and we do not need to use gap Diffie-Hellman group's algorithms such as pairing to solve the DDH problem. Since the TTP already knows its own secret key x_c, TTP can determine whether (g, R_1, y_c, R_2) is a DDH tuple by checking whether $R_2 = R_1^{x_c}$ holds.

Dispute Sub-protocol. If Bob has sent his signature σ_B to Alice but has not received R_2 or the received R_2 from Alice is invalid[6], he can invoke the dispute sub-protocol and sends to Charlie $(C_A, R_1, R', C, Dsc_m, ID_A, ID_B, ID_C, s_A, \sigma_B)$. Upon receiving the data from Bob, Charlie performs the following operations:

1. Charlie first validates the received data. This step is the same as the data validation in Steps 2 and 3 in the exchange sub-protocol. Charlie aborts if the validation fails. Otherwise, he continues.
2. Charlie computes $R_2 = R_1^{x_c} \in G$, and applies the decryption operation to obtain the message $m(= \mathcal{D}_{H_1(R_2)}(C))$. If m does match its description Dsc_m and $R' = H(R_2)$, Charlie sends R_2 to Bob and σ_B to Alice.

Remark 2.2. If m does match its description Dsc_m, or $R' \neq H(R_2)$, Alice cannot use Bob's receipt to prove to others that Bob has received the message m from her since the data validation described after Remark 2.1 would fail.

3 Security Discussion

Security of our certified email protocol is analyzed with the following three lemmas:

Lemma 3.1. *The modified Schnorr signature scheme is secure against the adaptively chosen message attack with the discrete logarithm (DL) assumption in random oracle model and public key substitute attack.*

Proof. Compared with the original Schnorr signature scheme [40], the only difference in the modified Schnorr signature scheme is $H(m||R||y)$ instead of $H(m||R)$. In random oracle model [14], however, both hash oracles can choose to respond with the same output to the query to $H(m||R||y)$ on input (m, R, y) and the

[5] Note that a gap Diffie-Hellman group is always a CDH-hard group but not vice versa.
[6] A received R_2 is considered as invalid if the decrypted message m does not match its description Dsc_m, or $R' \neq H(R_2)$.

query to $H(m||R)$ on input (m, R). Since the Schnorr signature scheme is proved to be secure against the adaptively chosen message attack with the DL assumption in random oracle model [33], we conclude that the modified Schnorr signature scheme is also secure against the adaptively chosen message attack in random oracle model. According to the security analysis of [7] [Section 5], on the other hand, the modified Schnorr signature scheme can resist the public key substitute attack, i.e., there exists only a negligible possibility that a different public key can be found to satisfy the signature corresponding to a specified public key.

As a result, we conclude that the lemma holds. □

Lemma 3.2. *Assume that the Computational Diffie-Hellman (CDH) assumption holds, and the hash function $H_2(\cdot)$ is a secure one-way hash function, then only Alice and Charlie can deduce the message encryption key k which is used to encrypt the message m in our certified email protocol.*

Proof. According to Lemma 3.1, only Alice can produce a valid signature (R_1, σ). In other words, R_1 is guaranteed to be generated by Alice, i.e., no one can impersonate Alice to send a valid R_1. Since the hash function $H_2(\cdot)$ is a secure one-way hash function, the only way to deduce the message encryption key k is to deduce the value of R_2. The CDH assumption implies that it is impossible to deduce R_2 from R_1 and y_s. Therefore, no one except the person who knows r or x_s can deduce the value of k. This means that only Alice and Charlie can deduce the message encryption key k. □

Lemma 3.3. *Our certified email protocol can provide fairness.*

Proof. Based on the description presented in the above section, when the exchange sub-protocol is executed normally, i.e., when Alice and Bob are honest and the communication channel works, Bob receives the message sent by Alice, Alice receives a receipt from Bob, and Charlie is not involved. What's more, if Alice and Bob are both honest, but the communication channel does not work during the execution of the exchange sub-protocol; Alice can invoke the dispute protocol to ask for TTP's help to complete the exchange. Therefore, fairness holds in these two cases. In other cases, we are going to show that our proposed protocol can also provide fairness, i.e., Alice and Bob cannot take advantage over the other in the process of execution of our protocol even if he or she behaves maliciously. Those cases are classified into the following three cases: (1) Alice is honest, but Bob is malicious; (2) Bob is honest, but Alice is malicious, and (3) Alice and Bob are both malicious.

Case 1: *Alice is honest, but Bob is malicious.* In this case, Bob aims to obtain the message m without sending his valid receipt σ_B to Alice. In our certified email protocol, Bob may cheat in Step 2 of the `exchange sub-protocol` by not sending his valid receipt to Alice. According to our protocol, however, Alice will not send the value R_2 to Bob in this situation. Bob can obtain R_2 from Charlie by executing the `dispute sub-protocol`. But in this case, he

has to send his valid receipt to Charlie before Charlie forwards R_2 to Alice. Charlie also forwards the receipt to Alice in the `dispute sub-protocol`. Furthermore, according to Lemma 3.1, only Alice can generate a valid signature (R_1, σ). In conclusion, if Bob wants to receive m, he has to send his valid receipt to Alice, directly or indirectly. Our protocol can provide fairness in this case.

Case 2: *Bob is honest, but Alice is malicious.* In this case, Alice aims to obtain Bob's receipt σ_B without sending the message m to Bob, or to make Charlie abort in `dispute sub-protocol`. In our protocol, Alice may cheat in two steps: Step 1 and Step 3 of the exchange sub-protocol. In the former one, if Alice does not send the authorized data[7] $(C_A, R_1, R', Dsc_m, s_A)$ to Bob, Bob will not send his valid receipt to Alice. On the other hand, if Alice does not send the right[8] (R_1, R') to Bob, but Bob would send the valid receipt to Alice. Alice cannot use the received receipt from Bob to prove to others that Bob has received the right message m from her, which means the receipt Alice received is useless. Therefore Alice has to send the authorized and right $(C_A, R_1, R', Dsc_m, s_A)$ to Bob in this step. In the latter one, if Alice sends invalid R_2 to Bob or does not sends R_2 to Bob, Bob can invoke the dispute sub-protocol to get m. If the received message m does not match its description, the receipt Alice obtains from Bob is useless since she cannot prove to others that Bob has received the right message m from her. In conclusion, our protocol can provide fairness in this case too.

As a result, we finish our proof. □

4 Efficiency

In this section, we compare our proposed protocol with others. To the best of our knowledge, all the existing optimistic certified email protocols are based on public key cryptography technologies. Public key cryptography takes much longer time than symmetric key cryptography or secure hash functions. In public key cryptography, the most time-consuming operation is the modular exponentiation calculation. In fact, the ratio of the time taken for a modular exponentiation operation to the time taken for a single modular multiplication is linearly proportional to the exponent's bit length [8]. As a result, we ignore single modular multiplications and other non-public key cryptography algorithms such as symmetric encryption, symmetric decryption, and hash function in our theoretical analysis of our protocol's efficiency and comparison with other certified email protocols.

To the best of our knowledge, the two protocols proposed by Wang [35,36] are the most efficient certified email protocols previously proposed with an off-line TTP. One protocol is based on the ElGamal scheme [22] and the Schnorr scheme

[7] Authorized data means others can make sure that the data is from Alice.

[8] Right means Charlie and Alice would result in the same symmetric encryption key k, and $R' = H(R_2)$.

Table 1. Comparison of time cost of our proposed protocol with Wang's

	Wan05a	Wan05b	Ours
Step 1 of Exchange	$3EXP$	$1EXP_{RSA} + 1EV_{RSA}$	$2EXP$
Step 2 of Exchange	$2EXP + 1SGN_B$ [a]	$1EV_{RSA} + 1SGN_B$	$2EXP + 1SGN_B$
Step 3 of Exchange	$1VER_B$ [b]	$1VER_B$	$1VER_B$
Total of Exchange	$5EXP+$ $1SGN_B + 1VER_B$	$1EXP_{RSA} + 2EV_{RSA}+$ $1SGN_B + 1VER_B$	$4EXP+$ $1SGN_B + 1VER_B$
Prove to Other	$2EXP$	$1EV_{RSA}$	$1Pairing$ [c] (or $1EXP$ [d])
Dispute	$3EXP + 1VER_B$	$1EXP_{RSA} + 1EV_{RSA} + 1VER_B$	$3EXP + 1VER_B$

[a] Time taken by Bob's signature algorithm.
[b] Time taken by Bob's verfication algorithm.
[c] Time taken by a pairing computation.
[d] In this case, Alice can only prove to Charlie.

[40] (denoted as Wan05a). The other is based on RSA (denoted as Wan05b). As a result, our protocol is compared with only these two protocols in efficiency comparison. We use EXP to denote the time taken by one modular exponentiation operation that ElGamal encryption scheme or the Schnorr scheme need. EXP_{RSA} denotes the time taken by one modular exponentiation operation that RSA signature or RSA decryption needs, and EV_{RSA} denotes the time taken by one modular exponentiation operation that RSA verification or RSA encryption needs. We assume that our proposed protocol uses the same group G as the group in Wan05a, no matter it is a multiplication group of a finite field or a finite rational point group over an elliptic curve.

Table 1 shows the time cost of our proposed protocol as well as Wang's protocols. The time costs in the setup phase and the certificate verification process are ignored. From the table, our protocol saves one modular exponentiation operation in the exchange sub-protocol as compared with Wan05a. If Alice needs to prove to only the TTP that she has sent the message m to Bob, one protocol saves one modular exponentiation operation in the proving process. If Alice needs to prove to others, then our protocol needs one pairing operation, which is typically slower than the two modular exponentiation operations needed in Wan05a. Comparison of our protocol with Wan05b is more complex due to different public key cryptography systems used in the two protocols. Wan05b uses RSA. ElGamal encryption scheme and the Schnorr signature scheme used in Wan05a and ours are based discrete logarithm problem, and, as a result, can take the advantage of Elliptic Curve Cryptography (ECC) which uses much shorter keys than, and therefore much faster than RSA for the same security level. For example in [41]: RSA with 2048 bits of key length has the same security level as ECC with 224 bits of key length, and ECC-224 is 7.8 times faster than RSA-2048 in full length modular exponentiation. Therefore, our protocol is also much more efficient than Wan05b. In conclusion, our protocol is more efficient than both Wan05a and Wan05b, the two most efficient certified email protocols with off-line TTP.

5 Conclusion

In this paper, we presented a novel approach to construct a certified email protocol. This new approach is based on the authorized Diffie-Hellman key agreement. Our proposed protocol is the most efficient certified email protocol among all the existing certified email schemes in terms of the number of exponentiations and communication data. Due to its efficiency, our certified email protocol is very suitable for applications in a distributed environment.

References

1. Ateniese, G.: Efficient Verifiable Encryption (and Fair Exchange) of Digital Signatures. In: ACM CCS 1999, pp. 138–146 (1999)
2. Abadi, M., Glew, N., Horne, B., Pinkas, B.: Certificated email with a light online trusted third party: Design and implementation. In: WWW 2002, pp. 387–395 (2002)
3. Ateniese, G., de Medeiros, B., Goodrich, M.: TRICERT: Distributed certified email schemes. In: NDSS 2001 (February 2001)
4. Ateniese, G., Nita-Rotaru, C.: Stateless-recipient certified Email system based on verfiable encryption. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 182–199. Springer, Heidelberg (2002)
5. Asokan, N., Shoup, V., Waidner, M.: Optimistic Fair Exchange of Digital Signatures. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 591–606. Springer, Heidelberg (1998)
6. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures. IEEE Journal on Selected Areas in Communication 18(4), 593–610 (2000)
7. Bao, F.: Colluding Attacks to a Payment Protocol and Two Signature Exchange Schemes. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 417–429. Springer, Heidelberg (2004)
8. Boneh, D.: Twenty years of attacks on the RSA cryptosystem. Notices of the American Mathematical Society (AMS) 46(2), 203–213 (1999)
9. Boneh, D., Durfee, G.: Cryptanalysis of RSA with Private Key d Less Than $N^{0.292}$. IEEE Trans. on Info. Theo. 46(4), 1339–1349 (2000)
10. Bao, F., Deng, R., Mao, W.: Efficient and Practical Fair Exchange Protocols. In: Proceedings of 1998 IEEE Symposium on Security and Privacy, pp. 77–85 (1998)
11. Boyd, C., Foo, E.: Off-line Fair Payment Protocols using Convertible Signatures. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 271–285. Springer, Heidelberg (1998)
12. Ben-Or, M., Goldreich, O., Micali, S., Rivest, R.L.: A fair protocol for signing contracts. IEEE Transactions on Information Theory 36(1), 40–46 (1990)
13. Boneh, D., Naor, M.: Timed Commitments. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 236–254. Springer, Heidelberg (2000)
14. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: ACM CCS 1993, pp. 62–73 (1993)
15. Bahreman, A., Tygar, J.D.: Certified electronic mail. In: Proceedings of the Internet Society Symposium on Network and Distributed System Security, pp. 3–19 (1994)
16. Coffey, T., Daiha, P.: Non-repudiation with mandatory proof of receipt. Computer Communication Review 26(1), 6–17 (1996)

17. Chen, L.: Efficient Fair Exchange with Verifiable Confirmation of Signatures. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 286–299. Springer, Heidelberg (1998)
18. Damgård, I.B.: Practical and provably secure release of a secret and exchange of signatures. Journal of Cryptology 8(4), 201–222 (1995)
19. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory 22, 644–654 (1976)
20. Dodis, Y., Reyzin, L.: Breaking and Repairing Optimistic Fair Exchange from PODC 2003. In: ACM DRM 2003, pp. 47–54 (2003)
21. Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. Communications of the ACM 28(6), 637–647 (1985)
22. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Infor. Theory 31, 469–472 (1985)
23. Even, S., Yacobi, Y.: Relations among public key signature schemes, Technical Report 175, Computer Science Department, Technion, Israel (1980)
24. Goldreich, O.: A simple protocol for signing contracts. In: McCurley, K.S., Ziegler, C.D. (eds.) Advances in Cryptology 1981 - 1997. LNCS, vol. 1440, pp. 133–136. Springer, Heidelberg (1999)
25. Garay, J., Jakobsson, M., MacKenzie, P.: Abuse-free optimistic contract signing. In: Wiener, M.J. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 449–466. Springer, Heidelberg (1999)
26. Haller, N.M.: The S/KEY one-time password system. In: Proceedings of the Internet Society Symposium on Network and Distributed Systems (1994)
27. Markowitch, O., Roggeman, Y.: Probabilistic non-repudiation without trusted third party. In: Proc. of 2nd Conference on Security in Communication Networks (SCN 1999), Amalfi, Italy (1999)
28. Markowitch, O., Saeednia, S.: Optimistic fairexchange with transparent signature recovery. In: Syverson, P.F. (ed.) FC 2001. LNCS, vol. 2339, pp. 339–350. Springer, Heidelberg (2002)
29. Nenadić, A., Zhang, N., Barton, S.: A Security Protocol for Certified E-goods Delivery. In: Proc. IEEE Int. Conf. Information Technology, Coding, and Computing (ITCC'04), pp. 22–28 (2004)
30. Okamoto, T., Pointcheval, D.: The gap-problems: a new class of problems for the security of cryptographic Schemes. In: Kim, K.-c. (ed.) PKC 2001. LNCS, vol. 1992, pp. 104–118. Springer, Heidelberg (2001)
31. Park, J.M., Chong, E., Siegel, H., Ray, I.: Constructing Fair-Exchange Protocols for E-Commerce Via Distributed Computation of RSA Signatures. In: PODC 2003, pp. 172–181 (2003)
32. Park, J.M., Ray, I., Chong, E.K.P., Siegel, H.J.: A Certified email Protocol Suitable for Mobile Environments. In: GLOBECOM 2003, pp. 1394–1398 (2003)
33. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 387–398. Springer, Heidelberg (1996)
34. Imamoto, K., Sakurai, K.: A certified email system with receiver's selective usage of delivery authority. In: Menezes, A.J., Sarkar, P. (eds.) INDOCRYPT 2002. LNCS, vol. 2551, pp. 326–338. Springer, Heidelberg (2002)
35. Wang, G.: Generic fair non-repudiation protocols with transparent off-line TTP. In: IWAP 2005, pp. 51–65 (2005)
36. Wang, G.: Generic Non-Repudiation Protocols Supporting Transparent Off-line TTP. Journal of Computer Security 14(5), 441–467 (2006)

37. Ray, I., Ray, I.: An Optimistic Fair Exchange E-commerce Protocol with Automated Dispute Resolution. In: Bauknecht, K., Madria, S.K., Pernul, G. (eds.) EC-Web 2000. LNCS, vol. 1875, pp. 84–93. Springer, Heidelberg (2000)
38. Ray, I., Ray, I., Natarajan, N.: An anonymous and failure resilient fair-exchange e-commerce protocol. Decision Support Systems 39, 267–292 (2005)
39. Borrell, J., Robles, S., Serra, J., Riera, A.: Securing the Itinerary of Mobile Agents through a Non-Repudiation Protocol. In: Security Technology, 1999. Proceedings. IEEE 33rd Annual 1999 International Carnahan Conference on, pp. 461–464 (1999)
40. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of Cryptology 4(3), 161–174 (1991)
41. Eberle, H., Gura, N., Shantz, S.C., et al.: A Public Key Cryptography processor for RSA and ECC. In: Application-Specific Systems, Architectures and Processors, pp. 98–110 (2004)