SCIS 2006 The 2006 Symposium on Cryptography and Information Security
Hiroshima, Japan, Jan. 17-20, 2006
The Institute of Electronics, Information and Communication Engineers

Optimistic Fair Exchange Protocol for E-Commerce

Yusuke Okada * Yoshifumi Manabe † Tatsuaki Okamoto †

Abstract— For e-commerce payments, fair exchange is one of the essential problems. The optimistic fair exchange protocol allows two parties to efficiently exchange items so that either each party gets the other's item or neither does. We propose a new optimistic fair exchange protocol that is efficient and applicable to any digital signature scheme such as RSA or DSA. In our protocol, we introduce *pre-signature*, *post-signature* and *notarized signature* by prescribing the form of the digital signatures. Furthermore, we introduce a parameter that represents the expiration date of the pre-signature to realize the timely termination of the protocol.

Keywords: Optimistic fair exchanges, e-commerce, digital signatures.

1 Introduction

Fair exchange is one of the essential problems in e-commerce. It allows two parties to either gets the other's item or neither does. For example, Alice (customer) wants to buy music file M_B from Bob (merchant) in exchange for her digital signature σ_A (digital cash, digital check, etc.) over the Internet. In such a case, Alice does not want to give her digital signature until she gets the music file from Bob. On the other hand, Bob wants Alice's digital signature before he passes the music file to Alice. It is difficult, however, to guarantee simultaneous exchanges over the Internet.

The most simple way to solve this dilemma is to use a *Trusted Third Party* (TTP). First, Alice and Bob send σ_A and M_B to TTP respectively. After making sure of both items, TTP sends M_B to Alice and σ_A to Bob. Such fair exchange protocols are quite simple, but since TTP mediates all transactions, TTP becomes a bottleneck, and the maintenance cost of TTP is very high.

To overcome this drawback, the *optimistic* fair exchange protocol was proposed. Since TTP is invoked only when either party does something wrong in the exchange protocol, the optimistic fair exchange protocol is more efficient than the protocol with online TTP in which TTP mediates all transactions.

The core primitive of optimistic fair exchange is *pre-signature* σ'. The pre-signature itself has no intrinsic value. Only the signer of the pre-signature and TTP have the power to transform pre-signature σ' to signature σ. The basic exchange protocol is as follows.

Exchange protocol

1. Alice sends her pre-signature σ'_A to Bob.
2. Bob verifies σ'_A and if valid, sends M_B to Alice.
3. After receiving M_B from Bob, Alice sends Bob her signature σ_A.
4. If Alice doesn't send σ_A, then Bob requests TTP to transform σ'_A into σ_A.

In this paper, we realize a pre-signature scheme by prescribing the form of the signatures. To date, many approaches have been taken to realize the pre-signature described above. Until Park et al. [14] proposed the optimistic fair exchange protocol based on two-party multisignature, most used zero knowledge proofs in the exchange stage. The zero knowledge proof protocols have been a highly interactive part of the exchange protocol. The protocol of [14] does not use any zero knowledge proofs in the exchange stage, only uses in the key registration stage and so is needed only once. However, it was broken by Dodis et al. [7]. They proposed a protocol utilizing the GDH-based multisignature of Boldyreva [4], but it requires special elliptic curve groups with a bilinear map. Our optimistic fair exchange protocol is more simple and efficient than these previous works, and can use any existing signature scheme such as RSA or DSA. Furthermore, we realize the timely termination of the protocol by introducing a parameter that represents the expiration date of the pre-signature.

2 Related work

Fair exchange protocols can be put into three categories: (I) gradual secret exchange protocols, (II) fair exchange protocols with online TTP, (III) fair exchange protocols with offline TTP (or optimistic fair exchange protocols).

2.1 Gradual secret exchange protocol

In gradual secret exchange protocols [8, 13], it is assumed that Alice and Bob have secrets a and b respectively, both of which are n bits long. The protocol is

* Department of Social Informatics, Graduate School of Informatics, Kyoto University, Yoshidahonmachi, Sakyo-ku, Kyoto-shi, Japan. E-mail: yokada@lab7.kuis.kyoto-u.ac.jp
† NTT Laboratories, NTT Corporation, 1-1 Hikarinooka, Yokosuka-shi, Japan. E-mail: {manabe.yoshifumi, okamoto.tatsuaki}@lab.ntt.co.jp

as follows. First, Alice and Bob exchange $f(a)$ and $g(b)$, where $f(\cdot)$ and $g(\cdot)$ are one-way functions so that Alice can't get b from $g(b)$ and Bob can't get a from $f(a)$. Next, Alice and Bob exchange their secrets bit by bit so that finally each gets the other's secret. These protocols do not need TTP, but are impractical because they are highly interactive and assume that both parties have equal computational power for calculating the other's remaining secret at any stage of the protocol. Considering B2C e-commerce, merchants (software vender, online music download store, etc.) often have much more computational power than customers who usually have only one or two personal computers.

Gradual secret exchange protocols exchange two secrets a and b, both n bits long, so if Alice and Bob want to exchange their signatures, Alice adds M_A a declaration statement that the document is valid only if Bob can show her secret a and Bob does the same. However, to exchange the digital data and the digital signatures (in our case), Bob encrypts M_B by his encryption key K and sets K as his secret b. But it is difficult for Alice to verify whether M_B is genuine or not before she decrypts and opens the file.

2.2 Fair exchange with online TTP

Fair exchange protocols with online TTP are relatively simple. These protocols use non-repudiation techniques [15] to capture fairness. ISO has standardized the non-repudiation techniques [9, 10, 11]. There are some drawbacks. TTP maintenance cost is expensive and TTP can become a bottleneck. So we may face the problem of scalability to use fair exchange protocols with online TTP.

2.3 Fair exchange with offline TTP

The optimistic fair exchange protocols can be put into three subcategories by their primitives used to realize fairness: (i) verifiable escrow [1], (ii) verifiably encrypted signature [2, 3], (iii) two-signature [7, 14].

Verifiable escrow Asokan et al. [1] proposed an optimistic fair exchange protocol that uses *verifiable escrow*. To use TTP as an escrow service, a signer encrypts his/her signature under the public key of TTP. Verifiable escrow is an encryption scheme with an attached decryption policy that represents the conditions under which the encryption will be decrypted by TTP. First, the signer reduces his/her signature to a certain homomorphic pre-image of the signature. The signer then verifiably escrows the homomorphic pre-image using a cut-and-choose interactive zero-knowledge proof. This scheme is applicable to any signature as long as the signature scheme can be reduced to a certain homomorphic pre-image of the signature. They introduced homomorphic pre-signatures of RSA, DSA, Schnorr, Fiat-Shamir signatures and so on. The drawback of this protocol is that it uses cut-and-choose techniques and so it is highly interactive and needs a high amount of computation.

Verifiably encrypted signature Bao et al. [3] proposed a fair exchange protocol with offline TTP that uses a new primitive *Certificate of Encrypted Message Being a Signature* (CEMBS). In this protocol, parties sign their messages (such as a contract) and encrypt their signatures. CEMBS is used to convince parties that an encrypted signature is a certain party's signature on a message without revealing the signature itself. To realize this property, CEMBS uses proof-of-knowledge techniques and has to utilize a combination of a particular public key cryptosystem and digital signature scheme (in [3], they use ElGamal and DSA, or ElGamal and Gullou-Quisquater). This *ad-hoc* technique is not a desirable property.

Boneh et al. [5] recently proposed a new verifiably encrypted signature scheme based on the GDH signature of [6]. This scheme is completely non-interactive, but needs special elliptic curve groups with a bilinear map.

Two-signature Park et al. [14] introduced the optimistic fair exchange protocol which use the two-party multisignature scheme as a primitive element. We use the term *two-signature* to represent two-party multisignature quoting from [7]. In [14], they composed a two-signature scheme based on RSA signature, but Dodis et al. [7] broke this scheme.

Recently, Boldyreva [4] proposed a non-interactive multisignature scheme based on the GDH signature of Boneh et al. [6]. Dodis et al. [7] introduced an optimistic fair exchange protocol by utilizing the non-interactive multisignature of Boldyreva. Their two-signature scheme of [7] is as follows.

Alice randomly chooses $g \in G,\ x,\ x_1 \in \mathbb{Z}_p$ and computes $x_2 = x - x_1 \bmod p,\ v = g^x,\ v_1 = g^{x_1}$. Alice's public key is $pk = (g, v),\ pk_1 = v_1$ and secret key is $sk = x,\ sk_1 = x_1$. Alice then sends $pk,\ pk_1$ and x_2 to TTP, who checks $v \stackrel{?}{=} v_1 g^{x_2}$ and sets x_2 as a secret arbitration key *ask*. Alice's signature is $\sigma_A = H(m)^x$ and pre-signature is $\sigma'_A = H(m)^{x_1}$, where $H(\cdot)$ is a secure hash function. TTP can transform σ'_A into σ_A by calculating $\sigma_A = \sigma'_A H(m)^{x_2}$.

The protocol of [7] has two drawbacks. First, TTP has to safely store as many secret arbitration keys as the number of users. Next, it requires special elliptic curve groups with a bilinear map and two-signature scheme can't use RSA (two-signature scheme based on RSA of [14] is insecure), which is used very widely.

3 Motivating example of exchange protocols

Credit card payment is the most popular way in B2C e-commerce. In current online credit card payment systems, we can buy goods only by filling in application forms (credit card number, expiration date, an address, price of goods, and so on) and we don't show the credit card to the merchant. So the credit card number is, in a sense, an unchanging password, and if a certain person's credit card number and related information are leaked, he/she is exposed to the danger that someone may illegally use this credit card at online shops. That is, the system is built on the customers' trust that mer-

chants hold the information securely. This is not desirable. There is another risk that we have the credit card numbers and related information stolen in the real world when we show the credit cards.

To avoid this, we suggest the framework in which the order form is valid only if the customer signs it using his/her secret key. By utilizing the digital signature, we can build a secure online credit card payment system given the customer's safekeeping of his/her secret key (e.g., the use of smart cards). In this framework, we can apply our optimistic fair exchange protocol to an online credit card payment system.

4 Optimistic fair exchange protocols

We represent here our optimistic fair exchange protocol which is applicable to any digital signature scheme such as RSA and DSA. The parties involved in this protocol are Alice (customer), Bob (merchant) and TTP. Alice has the signature σ_A on M_A that represents digital cash, digital check, or the signature on the application form (See Section 3. In this case, M_A is the application form itself.). Bob has the digital data M_B such as online software or a music file. We don't consider the digital data that allows Alice to get a benefit if she gets multiple copies of the same M_B. This assumption is required when the dispute resolution protocol is invoked.

Next, we define *pre-signature*, *post-signature*, and *notarized signature*. We then describe the exchange protocol, the dispute resolution protocol and parameter t. We will mention the security of this protocol in the next section.

4.1 Definition of the pre-signature, post-signature and notarized signature

In this section, we define *pre-signature*, *post-signature*, and *notarized signature* by prescribing the form of the signatures. These are set as a working policy. XML and XML signatures are well-suited to implement this scheme.

We can use any secure digital signature scheme, and we assume the signature scheme consists of the triple of algorithm $(KeyGen, Sign, Verify)$. We also assume that Alice and TTP have already generated their secret and public key pairs by $KeyGen$ in the setup, and used PKI to certify their public keys.

The definitions of *pre-signature*, *post-signature*, and *notarized signature* are as follows.

Pre-signature Alice's pre-signature is of the form

$$\sigma'_A = Sign_A(M_A, cert_A, cert_{TTP}, t).$$

Post-signature Alice's post signature is of the form

$$\sigma_A = Sign_A(M_A, cert_A).$$

Notarized signature The notarized signature by TTP is of the form

$$\sigma_{TTP} = Sign_{TTP}(\sigma'_A).$$

The terms $cert_A$ and $cert_{TTP}$ mean the certificate of Alice and TTP respectively, and parameter t is the expiration date of the pre-signature. We introduce this parameter to realize *timely termination* of the protocol (see Section 4.4). The pre-signature σ'_A of the form above has no intrinsic value, whereas the post-signature σ_A has legal value as a digital signature. TTP has the power to transform Alice's pre-signature to *notarized signature* σ_{TTP} that has the same legal value as σ_A by signing Alice's pre-signature with TTP's secret key.

We define both σ_A and σ_{TTP} as legally valid signatures. Even if Bob shows both σ_A and σ_{TTP}, these are regarded as one legally valid signature.

4.2 Description of the optimistic fair exchange protocol

We describe here details of the optimistic fair exchange protocol. Figure 1 shows the outline of the protocol. In the protocol, we assume the data transactions are executed over a secure channel established using a technique such as SSL. Alice initiates the protocol with Bob. The protocol is as described below.

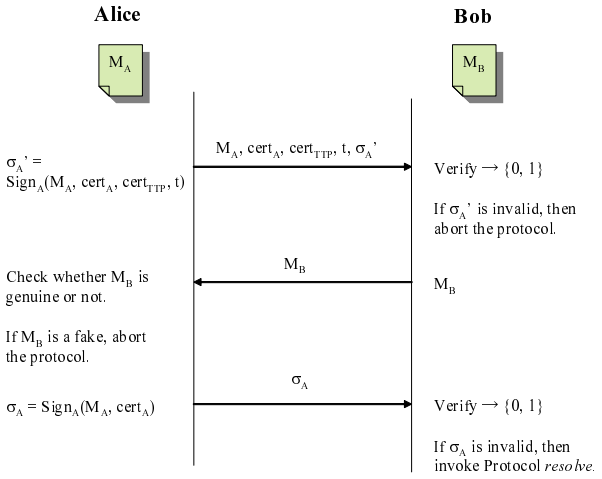

Figure 1: Overview of the protocol

1. Alice signs (M_A, $cert_A$, $cert_{TTP}$, t) using her secret key, and sends her pre-signature σ'_A to Bob.
2. Bob verifies σ'_A, and if σ'_A is invalid, Bob aborts the protocol.
3. If σ'_A is valid, then Bob sends M_B to Alice.
4. Alice receives M_B and checks whether M_B is genuine or not. If M_B is false or Alice doesn't receive M_B, Alice aborts the protocol.
5. After receiving the genuine M_B, Alice signs (M_A, $cert_A$) and sends her post-signature σ_A.
6. Bob verifies her post-signature σ_A. If σ_A is invalid or Bob don't get σ_A by the expiration date

of her pre-signature (parameter t), then Bob invokes dispute resolution protocol *resolve*.

7. If σ_A is valid, the exchange protocol ends correctly.

4.3 Dispute resolution protocol

We describe here the dispute resolution protocol *resolve* shown in Figure 2.

1. Bob initiates the protocol and sends Alice's pre-signature σ'_A, (M_A, $cert_A$, $cert_{TTP}$, t), and his digital data M_B to TTP.
2. TTP verifies Alice's pre-signature and M_B, and if either one of the two is invalid, TTP aborts protocol *resolve*.
3. If both σ'_A and M_B are valid, then TTP signs on σ'_A and sends the notarized signature σ_{TTP} to Bob.
4. TTP also forwards M_B to Alice. Protocol *resolve* thus ends correctly.

The reason why Bob has to send M_B in step 1 above and TTP forwards M_B to Alice in step 4 is to prevent Bob from obtaining the notarized signature σ_{TTP} but not sending M_B to Alice.

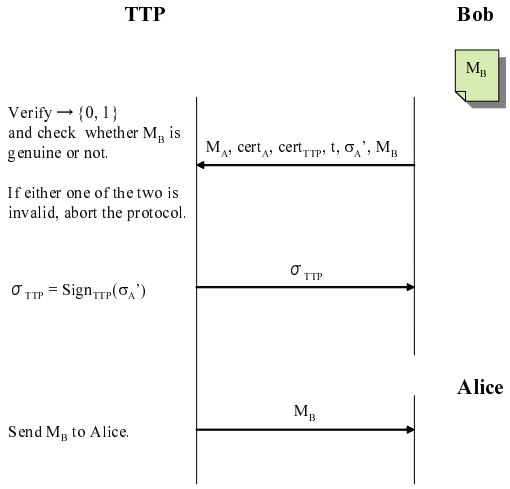

Figure 2: Protocol *resolve*

Verification of M_B

The TTP's verification of M_B may be a bottleneck, because it is difficult to associate the verification of M_B with a certain mathematical algorithm. To efficiently verify the digital data, we propose to use hash tables. We assume that there exists a hash table of music files or online software. TTP generates the message digest h_B of M_B by the hash function, and verifies M_B by checking whether h_B is in the hash table or not.

4.4 Parameter t

Parameter t defines the expiration date of Alice's pre-signature, and TTP won't transform σ'_A after this expiration date. This parameter may be set by Alice or set as a system parameter, and is essential for the *timely termination* of the exchange protocol. This is because both Alice and Bob will be at a disadvantage if they set an unfavorable t or don't terminate the protocol by the expiration date.

Disadvantage for Alice Even if Alice sets an unfavorable t (e.g., too short, or past date), Bob wouldn't send M_B, so Alice can't get any advantage from this. Besides, we can avoid this condition by setting parameter t as a system parameter.

Disadvantage for Bob If Bob sends M_B after the expiration date of Alice's pre-signature and Alice doesn't send her post-signature, Bob can not get Alice's valid signature. So parameter t constrains Bob to send M_B by the expiration date. If Bob sends M_B and Alice doesn't send σ_A until the expiration date, Bob invokes protocol *resolve* and has Alice's pre-signature σ'_A transformed into the notarized signature σ_{TTP} by the expiration date.

For practical purposes, we should prearrange when Bob can invoke protocol *resolve*. That is, for example, Bob will be able to have σ'_A transformed into σ_{TTP} between the expiration date and the following day. Bob invokes protocol *resolve* during this period.

5 Security analysis

In this section, we analyze the security of our optimistic fair exchange protocol. We assume that digital signature schemes used in the protocol are secure, so we analyze whether Alice and Bob are assured of *fairness*.

Fairness means either each party gets the other's item or neither does at the end of the protocol. In other words, no dishonest party can get the honest player's item without the honest player getting the dishonest player's item.

Security of Alice If Alice receives false M_B or doesn't receive M_B after sending σ'_A, Alice won't send her post-signature and aborts the protocol. Bob gets σ'_A, and in order to transform σ'_A into σ_A without TTP, Bob has to forge the signature. Because we assume that the digital signature schemes used in the protocol are secure, Bob can't forge signatures so neither gets any intrinsic value.

Then, if Bob doesn't send M_B and asks TTP to transform σ'_A into σ_{TTP}, Bob must send M_B to TTP. So, Alice can get M_B from TTP and Alice and Bob gets M_B and σ_{TTP} respectively.

Bob can get both σ_A and σ_{TTP} by invoking protocol *resolve* after receiving σ_A. However, the possession of σ_A and σ_{TTP} is regarded as the possession of one legally valid signature (see Section 4.1). So fairness is assured in this case.

Security of Bob First, if Alice doesn't send σ'_A or sends an invalid pre-signature, Bob aborts the protocol, neither gets any intrinsic value. Second, if Bob receives an invalid σ_A or doesn't receive σ_A after sending M_B, Bob invokes protocol *resolve* and has σ'_A transformed into σ_{TTP}.

6 Conclusion

In this paper, we presented a new optimistic fair exchange protocol. We prescribed three forms of signatures: pre-signature, post-signature, post-signature and notarized signature as primitives that realize fairness. By using these primitives, our protocol is computationally very efficient and any existing digital signature schemes can be used. Furthermore, we introduced parameter t, which represents the expiration date of pre-signatures; it realizes timely termination of the protocol.

References

[1] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communication*, volume 18, No. 4, pages 593-610, 2000.

[2] G. Ateniese. Efficient verifiable encryption (and fair exchange) of digital signatures. In *Proceedings of the 6th ACM conference on Computer and communications security*, pages 138-146, 1999.

[3] F. Bao, R. Deng, and W. Mao. Efficient and practical fair exchange protocols with off-line TTP. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 77-85, 1998.

[4] A. Boldyreva. Efficient threshold signature, multisignature and blind signature schemes based on the Gap-Diffie-Hellman-group signature scheme. In *Proceedings of Practice and Theory in Public Key Cryptsystems - PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 31-46. Springer-Verlag, 2003.

[5] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 416-432. Springer-Verlag, 2003.

[6] D. Boneh, B. Lynn, and H. Shacham. Short signatures from weil pairing. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514-532. Springer-Verlag, 2001.

[7] Y. Dodis and L. Reyzin. Breaking and repairing optimistic fair exchange from PODC 2003. In *Proceedings of the 2003 ACM workshop on Digital rights management*. pages 47-54, 2003.

[8] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, volume 28, No. 6, pages 637-647, 1985.

[9] ISO/IEC DIS 13888-1. Information technology - Security techniques - Non-repudiation - Part 1: General model. ISO/IEC JTC1/SC27N1503, November 1996.

[10] ISO/IEC 5th CD 13888-2. Information technology - Security techniques - Non-repudiation - Part 2: Using symmetric techniques. ISO/IEC JTC1/SC27N1505, November 1996.

[11] ISO/IEC DIS 13888-3. Information technology - Security techniques - Non-repudiation - Part 3: Using asymmetric techniques. ISO/IEC JTC1/SC27N1507, November 1996.

[12] H. Maruyama, T. Nakamura, and T. Hsieh. Optimistic fair contract signing for Web services. In *Proceedings of the 2003 ACM workshop on XML security*, pages 79-85, 2003.

[13] T. Okamoto and K. Ohta. How to simultaneously exchange secrets by general assumptions. In *Proceedings of 2nd ACM Conference on Computer and Communications Security*, pages 184-192, 1994.

[14] J. M. Park, E. Chong, and H. J. Siegel. Constructing fair-exchange protocols for E-commerce via distributed computation of RSA signatures. In *Proceedings of the twenty-second annual symposium on Principles of distributed computing*, pages 172-181, 2003.

[15] J. Zhou and D. Gollmann. A fair non-repudiation protocol. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 55-61, 1996.