NEW FAIR PAYMENT PROTOCOLS

Hao Wang, Heqing Guo, Manshan Lin
School of Computer Science & Engineering, South China University of Tech., Guangzhou, China

Keywords: Electronic payment, Fairness, Abuse-freeness, Invisible TTP, RSA signature

Abstract: Fair payment protocol is designed to guarantee fairness in electronic purchasing, that is, no party can falsely deny involvement in the transaction or having sent/received the specific items/payment. In this paper we first present an efficient fair payment protocol providing invisibility of TTP, timeliness, and standard RSA signatures as the final non-repudiation evidences. Then we present a second payment protocol which is the first one to provide abuse-freeness. Our protocols can be easily integrated into the existing electronic payment systems.

1 INTRODUCTION

Fair payment protocol is designed to guarantee fairness in electronic purchasing, that is, no party can falsely deny involvement in the transaction or having sent/received the specific items/payment. For simplicity, we assume the merchant (Alice) and the client (Bob) have settled on the item and the price. Then the fair payment protocol only needs to deal with the exchange the item with the receipt signed by Bob. The receipt includes item identity, price information, Bob's account information, etc. So with the signed receipt, Alice can get her payment from Bob's account through the bank. Besides the item, Alice also needs to send to Bob a non-repudiation evidence of origin (NRO) proving she has sent the item. The receipt Bob generates can be seen as a non-repudiation evidence of receipt (NRR). The protocol must assure that both parties either get his/her expected item/receipt/evidence or nothing.

Asokan, Schunter, & Waidner (Apr. 1997) introduces the idea of optimistic approach and presents fair protocols with offline TTP, in which TTP intervenes only when an error occurs (network error or malicious party's cheating). Ever since then, subsequent efforts in this approach resulted in efficient and fair protocols (Asokan & Shoup (1998), Kremer, & Markowitch (May 2000), we call them as *AK protocol*) that can guarantee that both parties can terminate the protocol timely while assuring fairness (called property of *timeliness*). Their messages & rounds optimality and basic building blocks (main protocol, resolve and abort sub-protocols) are well analyzed and widely accepted. *Invisible TTP* is first introduced by Micali Micali (1997) to solve this problem. The TTP can generate exactly the same evidences as the sender or the recipient. In this way, judging the outcome evidences and received items cannot decide whether the TTP has been involved. There are two way of thinking: 1)The first one is using *verifiable signature encryption* (VSE). It means to send the signature's cipher encrypted with TTP's public key before sending the signature itself. And try to convince the recipient that it is the right signature and it can be recovered (decrypted) by TTP in case of errors. But as Boyd & Foo (1998) has pointed out, verifiable encryption is computationally expensive. 2) The other approach is to use *convertible signatures* (CS) and it is recently focused approach. It means to firstly send a partial committed signature (verifiable by the recipient) that can be converted into a full signature (that is a normal signature) by both the TTP and the signer. Works by Boyd & Foo (1998) and Markowitch & Kremer (Oct. 2001) are early efforts to use this approach to construct fair protocols.

Abuse-freeness, as a new requirement of fair protocols, is first mentioned by Boyd & Foo (1998), and formally presented by Garay, Jakobsson & MacKenzie (1999). And Garay et al. have also realized an abuse-free contract signing protocol. Based on the Jakobsson-Sako-Impagliazzo designated verifier signature (Jakobsson, Sako & Impagliazzo (1996)), they introduce a new signature scheme called

Wang H., Guo H. and Lin M. (2004).
NEW FAIR PAYMENT PROTOCOLS.
In *Proceedings of the First International Conference on E-Business and Telecommunication Networks*, pages 199-203
DOI: 10.5220/0001399101990203
Copyright © SciTePress

Private Contract Signature to realize this property.

Previous efforts studying the fairness issue in payment systems include Asokan, Schunter, & Waidner (Apr. 1997) and Boyd & Foo (1998). As discussed earlier, these two protocols are not efficient and practical enough as to recent advances in area of fair exchange: 1) the former one is based on the approach of VSE, which is computationally expensive and the generated evidences are not standard signatures; 2) the latter one suffers from relatively heavy communication burden for it involves an interactive verification process (in open network, less communication rounds means less risks), and it has taken little account of the important properties of timeliness and abuse-freeness.

In this paper we first present an efficient fair payment protocol with a non-interactive verification process. Then we present a second payment protocol which is the first one to provide abuse-freeness. And our new abuse-free protocol does not need any specialized signature scheme like Garay et al. (propose the Private Contract Signature), which achieve better simplicity and efficiency. When building our protocols, we also follow the principles proposed by Gurgens, Rudolph & Vogt (Oct. 2003) to strengthen security of protocol label and messages. The two protocols use an adaptation of the convertible signature scheme proposed by Gennaro, Krawczyk & Rabin (1997) (GKR scheme) and used by Boyd & Foo (1998). The original scheme uses an interactive verification process that is not practical for fair protocols. So we replace it by a non-interactive verifying approach developed by ourselves. That is how we construct the first protocol of our proposals. Using the designated verifier proofs by Jakobsson, Sako & Impagliazzo (1996) (and strengthened by Wang (2003)), we achieve abuse-freeness in our second protocol. Briefly, designated verifier proof means that the proofs can convince nobody except the designated verifier (say Bob) and its underlying statement is "Either θ is true or I can sign as Bob". In this way, outside parties will not believe θ is true as Bob can simulate this proof himself. In section 2, we present 5 important requirements for fair payment protocol and other preliminaries for later description use. Section 3 presents the first protocol and its detailed analysis. And section 4 describes how designated verifier proofs can be fixed into the scheme to build the abuse-free protocol. Section 5 gives some concluding remarks.

2 PRELIMINARIES

2.1 Requirement on Fair Payment Protocols

Based on these former works, we present a complete set of requirement definitions for fair payment protocols.

Definition 1 Effectiveness
A fair protocol is *effective* if (the communication channels quality being fixed) there exists a successful payment exchange for the payer and the payee.

Definition 2 Fairness
A fair protocol is *fair* if (the communication channels quality being fixed) when the protocol run ends, either the payer gets his/her expected goods and the payee gets the payment or neither of them gets anything useful.

Definition 3 Timeliness
A fair protocol is *timely* if (the communication channels quality being fixed) the protocol can be completed in a finite amount of time while preserving fairness for both exchangers.

Definition 4 Non-repudiability
A fair protocol is *non-repudiable* if when the exchange succeeds, either payer or payee cannot deny (partially or totally) his/her participation.

Definition 5 Invisibility of TTP
A fair protocol is *TTP-invisible* if after a successful exchange, the result evidences of origin/receipt and exchanged items are indistinguishable in respect to whether TTP has been involved.

2.2 Notation

We use following notation to describe the protocols.

- $h()$: a collision resistant one-way hash function
- $E_k()/D_k()$: a symmetric-key encryption/decryption function under key k
- $E_X()/D_X()$: a public-key encryption/ decryption function under pk_X
- $S_X()$: ordinary signature function of X
- $PS_X()$: partial signature function of X
- $FS_X()$: the final signature function of entity X
- k: the session key A uses to cipher *item*
- *receipt*: the receipt destined for A, it contains transaction id, item id, price information, B's account information, etc
- pk_X/ sk_X: public/secret key of X
- $cipher = E_k(item)$: the cipher of *item* under k
- l: a label that uniquely identifies a protocol run,

and it's computed by Alice: *l*=*h*(*A*, *B*, *TTP*, *h*(*cipher*), *h*(*k*))

- *f*: a flag indicating the purpose of a message

3 A FAIR PAYMENT PROTOCOL WITH INVISIBLE TTP

A register sub-protocol is presented because both parties must negotiate with TTP on some common parameters like shared secret keys. The registration sub-protocol between the Alice/Bob and TTP needs to be run only once. And the resulting common parameters can be used for any number of transactions.

Our protocol is based on an adapted version of the GKR signature scheme. Our adaptation makes major change based on the one by Boyd & Foo (1998). We mainly adapt the verification process and the basic idea is just the same. For completeness of description, we still give a sketch of the basic information and the registration sub-protocol.

Let *n* be the Alice's RSA modulus. *n* is a strong prime and it satisfies *n*=*pq* where *p*=2*p'*+1 and *q*=2*q'*+1 (*p*,*q*,*p'*,*q'* are primes). Her public key is the pair (*e*,*n*) and private key is *d*. To make the signature convertible, *d* is multiplicative divided in d_1 and d_2, satisfying $d_1 d_2 e = 1 \bmod \phi(n)$. d_1 (chosen by the TTP) is the secret key shared between Alice and TTP, it will be used to convert the partial signature to a final one.

Registration Sub-protocol

Alice requests for key registration by sending her public key pair (*e*, *n*) to the TTP. TTP checks the validity of *n* (by checking its certificate, the checking is denoted by *check_pk*()), if passes, it sends d_1 to Alice (for security, d_1 should be encrypted some way). Then Alice chooses a reference message ω (we choose 2 here) and computes $PS(\omega)=\omega^{d_2}$ and send them to TTP. TTP will check (using the function denoted by *check*ω ()) whether $\omega \equiv PS(\omega)^{d_1 e} \pmod n$ .If it holds, he will send a certificate $cert_A=S_{TTP}(A, e, n, \omega, PS(\omega))$ to Alice.

```
Registration Sub-protocol
1) $A \rightarrow TTP$: $f_{\mathbf{Reg}}$, $TTP$, $pk_A$
   if not check_pk() then stop
2) $TTP \rightarrow A$: $f_{\mathbf{Share}}$, $A$, $E_A(d_1)$
3) $A \rightarrow TTP$: $f_{\mathbf{Ref}}$, $\omega$, $PS(\omega)$
   if not check$\omega$ () then stop
4) TTP$\rightarrow A$: $f_{\mathbf{cert}}$, $A$, $cert_A$
```

With the certificate, Bob can be convinced that TTP can convert the partial signatures once they are signed by the same d_2 as $PS(\omega)$. Bob also need to involve such a registration sub-protocol to get his own certificate $cert_B$. Note that they may send the same reference message (we can safely use 2 here) to the TTP, which won't affect the security of the verification process.

Main Protocol

In this scheme, the partial signature is defined as $PS(m) = m^{d_2} \pmod n$ and it is converted to be the final signature using $FS(m) = PS(m)^{d_1} \pmod n$ It works because $FS(m)^e \equiv PS(m)^{d_2 e} \equiv m^{d_1 d_2 e} \equiv m \pmod n$ hold s.

Following we focus our attention on our non-interactive verification process. We assume that Alice knows $PS_B(\omega)$ and Bob knows $PS_A(\omega)$.

Generating Proofs

Alice selects a random number $u \in Z_q$, where q is large enough and publicly accessible and calculates

$$\begin{cases} \Omega = \omega^u \pmod n \\ M = m^u \pmod n \\ v = h(\Omega, M) \\ r = u + vd \pmod q \end{cases}$$

In this way, the proof of the *PS*(*m*), denoted by *pf*(*PS*(*m*)), is (*r*, Ω , *M*).

Verifying Proofs

When Bob gets the *PS*(*m*) and *pf*(*PS*(*m*)), he calculates

$$v = h(\Omega, M) \text{ and verifies} \begin{cases} \Omega PS(\omega)^v = \omega^r \pmod n \\ MPS(m)^v = m^r \pmod n \end{cases}$$

We denote the verifying operations as the function $verify(pf(PS_A(m)), m, PS_A(m), \omega, PS_A(\omega))$. If the verification fails, the function returns **false**.

In our protocol, we denote the content to be signed as *a*=($f_{\mathbf{NRO}}$, *B*, *l*, *h(k)*, *cipher*, $E_{TTP}(k)$), then the partial evidence of origin (**PEO**) is $PS_A(a)$ plus *pf* ($PS_A(a)$) and the final evidence, i.e. **NRO**, is $FS_A(a)$. Similarly, let *b*=($f_{\mathbf{NRR}}$, *A*, *l*, *receipt*), then the partial evidence of receipt (**PER**) is $PS_B(b)$ plus *pf* ($PS_A(b)$) and the **NRR** is $FS_B(b)$. After the first move, Bob needs to verify the Alice's partial signature. Bob will quit the protocol when the verification fails. When Alice gets the **PER**, she does the same thing. But instead of simply quitting the protocol, she needs to involve the abort sub-protocol. This is important because it will prevent Bob's later recovery that may result in unfair situation for Alice.

Main Protocol

1) $A \rightarrow B$: $f_{\mathbf{PEO}}$, B, l, $h(k)$, *cipher*, $E_{TTP}(l,k)$, $PS_A(a)$, $pf\,(PS_A(a))$

if not $verify(pf\,(PS_A(a)), a, PS_A(a), \omega, PS_A(\omega))$ **then** stop

2) $B \rightarrow A$: $f_{\mathbf{PER}}$, A, l, $PS_B(b)$, $pf\,(PS_A(b))$

if A times out **then abort**

if not $verify(pf\,(PS_B(b)), b, PS_B(b), \omega, PS_B(\omega))$ **then abort**

3) $A \rightarrow B$: $f_{\mathbf{NRO}}$, B, l, k, $FS_A(a)$

if B times out **then recovery**[X:=B,Y:=A]

4) $B \rightarrow A$: $f_{\mathbf{NRR}}$, A, l, $FS_B(b)$

if A times out **then recovery**[X:=A,Y:=B]

Recovery Sub-protocol

The recovery sub-protocol is executed when an error happens, one party needs TTP's help to decrypt the key k and generate the final evidences for him/her. The recovery request is denoted by $\mathbf{Rec}_X = S_X(f_{\mathbf{Rec}X}, Y, l)$.

Recovery Sub-protocol

1) $X \rightarrow TTP$: f_{RecX}, f_{Sub}, Y, l, $h(cipher)$, h(k), $E_{TTP}(l,k)$, $\mathbf{Rec}_{\mathbf{X}}$, $PS_A(a)$, $PS_B(b)$

if $h(k) \neq h(D_{TTP}(E_{TTP}(k)))$ **or** *aborted* **or** *recovered* **then** stop

else *recovered*=true

calculates $FS_A(a) = PS_A(a)^{d_{1A}} \pmod n$ and $FS_B(b) = PS_B(b)^{d_{1B}} \pmod n$

2) $TTP \rightarrow A$: $f_{\mathbf{NRR}}$, A, l, $FS_A(a)$

3) $TTP \rightarrow B$: $f_{\mathbf{NRO}}$, B, l, k, $FS_B(b)$

Abort Sub-protocol

Alice submits an abort request using abort sub-protocol, preventing Bob may recover in a future time which she will not wait. The abort request is denoted by **Abort**=$S_A(f_{\mathbf{Abort}}, TTP, l)$ and the abort confirmation is denoted by $\mathbf{Con}_a$= $S_{TTP}(f_{\mathbf{con}a}, A, B, l)$.

Abort Sub-protocol

1) $A \rightarrow TTP$: $f_{\mathbf{Abort}}$, l, B, **abort**

if *aborted* **or** *recovered* **then** stop

else *recovered*=true

2) $TTP \rightarrow A$: $f_{\mathbf{Con}a}$, A, B, l, $\mathbf{Con}_a$

3) $TTP \rightarrow B$: $f_{\mathbf{Con}a}$, A, B, l, $\mathbf{Con}_a$

Analysis of the Protocol

We can prove that our protocol satisfies all the requirements defined in section 2.1. (Proof is omitted for limited space).

4 MODIFY THE PROTOCOL TO PROVIDE ABUSE-FREENESS

In Section 3, we have used a secure non-interactive zero-knowledge proof to achieve an efficient fair payment protocol. But this kind of protocol may result in undesirable circumstances: because the partial signature's proof is universally verifiable, a not so honest Alice can present Bob's partial signature to an outside company proving that Bob has purchased something, and in this way to affect the company's purchasing decision. *Abuse-freeness*, firstly defined by Garay, Jakobsson & MacKenzie (1999). And Chadha, Mitchell, Scedrov & Shmatikov (Sep. 2003) propose a more precise definition of this property: one party couldn't prove to an outside party that the other party has participated in the protocol. The original verification algorithm by Jakobsson et al. works for an ElGamal-like public-key encryption scheme. So we replace the public generator g with ω in our protocol to adapt this verification. And we assume that Alice knows $PS_B(\omega)$ and Bob knows $PS_A(\omega)$.

Generating Proofs X selects $\alpha, \beta, u \in Z_q$ and calculates

$$\begin{cases} s = \omega^{\alpha} PS_Y(\omega)^{\beta} \bmod n \\ \Omega = \omega^{u} \bmod n \\ M = m^{u} \bmod n \\ v = h(s, \Omega, M) \\ r = u + d_X(v + \alpha) \bmod q \end{cases}$$

The proof of the $PS_X(m)$, denoted by $pf(PS_X(m))$, is (α , β , Ω , M, r).

Verifying Proofs When Y gets the $PS_X(m)$ and $pf(PS_X(m))$, s/he will calculate

$$\begin{cases} s = \omega^{\alpha} PS_Y(\omega)^{\beta} \bmod n \\ v = h(s, \Omega, M) \end{cases} \text{and}$$

$$\text{verifies} \begin{cases} \Omega PS_X(\omega)^{h+\alpha} = \omega^{r} \bmod n \\ MPS_X(m)^{h+\alpha} = m^{r} \bmod n \end{cases}$$

Simulating transcripts Y can simulate correct transcripts by selecting $t < n^2, \gamma < n^2, \eta < n^2$ and calculate

$$\begin{cases} s = \omega^{\gamma} \bmod n \\ \Omega = \omega^{t} PS_{X}(\omega)^{-\eta} \bmod n \\ M = m^{t} PS_{X}(m)^{-\eta} \bmod n \\ v = h(s, \Omega, M) \\ \mu = \eta - h \bmod q \\ r = (\gamma - \mu) d_{Y}^{-1} \bmod q \end{cases}$$

So Y cannot convince any outside party of the validity of the partial signature of Alice. We note these verifying operations as the function $verify(pf(PS_X(m)), m, PS_X(m), \omega, PS_X(\omega))$. If the verification fails, it returns **false**.

5 CONCLUSIONS

In this paper, we produce two fair payment protocols providing invisibility of TTP. They use the RSA-based convertible signature scheme. To be more efficient and practical for asynchronous network, we replace the original interactive verification process with a non-interactive one. To achieve abuse-freeness in the second protocol, we use an adaptation of the designated verifiers proofs by Jakobsson et al.

We have shown that these two protocols are practical because they meet all important requirements, their evidences are standardized, and the communication burden is row in that they only needs 4 interactions in a faultless run. Our future work will be focused on the application of the fair protocols to real electronic commerce systems like SCM and CRM.

REFERENCES

Asokan, N., Schunter, M., & Waidner, M. (Apr. 1997). Optimistic protocols for fair exchange. *Proceedings of the fourh ACM Conference on Computer and Communications Security, Zurich, Switzerland, ACM Press,* 6, 8-17.

Asokan, N., & Shoup, V. (1998). Optimistic fair exchange of digital signatures. *Advances in Cryptology -- EUROCRYPT '98, Berlin Germany, Lecture Notes in Computer Science, Volume 1403, Springer-Verlag,* 591-606.

Boyd, C. & Foo, E. (1998). Off-line Fair Payment Protocols using Convertible Signatures. *Advances in Cryptology---ASIA CRYPT'98.*

Chadha, R., Mitchell, J., Scedrov, A., & Shmatikov, V. (Sep. 2003). Contract signing, optimism and advantage. *CONCUR 2003 - Concurrency Theory, 14-th International Conference, Marseille, France, Lecture Notes in Computer Science, Volume 2761, Springer-Verlag,* 366-382.

Garay, J., Jakobsson, M., & MacKenzie, P. (1999). Abuse-free optimistic contract signing. *Advances in Cryptology - CRYPTO '99, Lecture Notes in Computer Science, Volume 1666, Springer-Verlag,* 449-466.

Gurgens, S., Rudolph, C., & Vogt, H. (Oct. 2003). On the Security of Fair Non-repudiation Protocols. *Proceedings of 2003 Information Security Conference, Bristol, UK, Volume 2851, Lecture Notes in Computer Science, Springer-Verlag,* 193-207.

Jakobsson, M., Sako, K., & Impagliazzo, R. (1996). Designated verifier proofs and their applications. *Eurocrypt'96, Volume 1070, Lecture Notes in Computer Science, Springer-Verlag,* 143-154.

Kremer, S., & Markowitch, O. (May 2000). Optimistic non-repudiable information exchange. *21th Symposium on Information Theory in the Benelux, Werkgemeenschap Informatie- en Communicatietheorie, Enschede,* 139-146.

Markowitch, O., & Kremer, S. (Oct. 2001). An optimistic non-repudiation protocol with transparent trusted third party. *Information Security: ISC 2001, Malaga, Spain, volume 2200, Lecture Notes in Computer Science, Springer-Verlag,* 363-378.

Micali, S. (1997). Certified e-mail with invisible post offices. *Available from author: an invited presentation at the RSA'97 conference.*

Saeednia, S., Kremer, S., & Markowitch, O. (Nov. 2003). An efficient strong designated verifier scheme. *6th International Conference on Information Security and Cryptology (ICISC 2003), Lecture Notes in Computer Sciences, Springer-Verlag. Seoul, Korea. November 2003.*

Wang, G. (2003). An Attack on Not-interactive Designated Verifier Proofs for Undeniable Signatures. *Cryptology ePrint Archive, Report 2003/243.* Retrieved Apr. 2004, from http://eprint.iacr.org/2003/243/.

Gennaro, R., Krawczyk, H., & Rabin, T. (1997). RSA-based undeniable signatures. *Advances in Cryptology --- CRYPTO '97, volume 1296, Lecture Notes in Computer Science, Springer Verlag,* 132--149.