AN OPTIMISTIC FAIR EXCHANGE PROTOCOL FOR TRADING ELECTRONIC RIGHTS

Masayuki Terada
Network Management Development Dept., NTT DoCoMo
te@rex.yrp.nttdocomo.co.jp

Makoto Iguchi
Information Sharing Platform Labs., NTT
iguchi@isl.ntt.co.jp

Masayuki Hanadate
Information Sharing Platform Labs., NTT
hanadate@isl.ntt.co.jp

Ko Fujimura
Information Sharing Platform Labs., NTT
fujimura@isl.ntt.co.jp

Abstract Reliable electronic commerce systems must offer fairness. In this paper, we propose a fair exchange protocol for trading electronic vouchers, which are the representation of rights to claim goods or services. This protocol enables two players to exchange vouchers stored in their smartcards fairly and efficiently. The players can exchange vouchers through a 4-round mutual communication protocol between their smartcards as long as the protocol is performed properly. If the protocol becomes unable to proceed due to misbehavior of the partner or for any other reason, the player that has fallen into the unfair condition (i.e. the player who sent its voucher but didn't receive the desired voucher) can recover fairness by performing a recovery protocol with a trusted third-party. Since the recovery protocol doesn't need the cooperation of the partner, fairness can be recovered without identifying or tracking the partner; one can trade vouchers securely even if the trading partner cannot be identified.

1. Introduction

Fairness is essential for realizing secure electronic commerce in which both consumers and merchants can participate with a sense of safety. From the consumers' viewpoint, payment should not be committed without receiving the merchandise purchased, while the merchants may not want to send the goods or services prior to being paid.

Both of the above requirements can be achieved easily at face-to-face transactions in real shops. However, they are difficult in electronic commerce because payments and the delivery of merchandise are rarely conducted simultaneously.

Once people are disadvantaged, they need to identify the trading partner and acquire compensation. Unfortunately, identifying the partner is not easy in the Internet[4]. Since this difficulty is likely to become even worse in the consumer-to-consumer (C2C) market that is forming, security that depends on an identification process is impractical.

In order to realize secure electronic commerce systems that dispense with identification, we focus on "electronic rights", irreproducible and transferable digital data representing the rights to claim services or goods[14]. Their real world equivalents are pieces of paper, i.e. tickets, coupons and vouchers. Following RFC3506[5], we refer to such data as (electronic) vouchers hereafter. Since electronic vouchers must be transferred among users and redeemed in an off-line environment like real tickets and coupons, we assume that they are stored and managed in tamper-resistant devices like smartcards to prevent illegal acts like reproduction and forgery.

A voucher can represent diverse types of rights. For instance, it can represent the right to claim goods like "one hamburger", or to claim services like "one night accommodation". Moreover, it can be used as it were money by representing the right to claim conversion into currency or a certain amount of some valuable item (e.g. gold); electronic money can be treated as a sort of voucher. The fair exchange of vouchers, therefore, enables us to realize not only secure barter trading but also secure purchase transactions[7].

Identification of the trading partner is not required in these transactions provided that the fairness of the exchange is guaranteed. The only thing they need to certify is the genuineness of the vouchers being exchanged. Transaction security, therefore, doesn't depend on the trustworthiness of the trading partner, but rather on that of the voucher's issuer. This simplifies the certification process because there would be many fewer voucher issuers than participants. Fair exchange of vouchers is thus the key component to realize fair and secure electronic commerce while dispensing with the need for identification[6].

Up to now, however, there has been no efficient method to exchange vouchers with fairness. A mediating (or active) trusted third-party (TTP) can exchange vouchers fairly, but this approach has drawbacks in terms of availability and

scalability because the TTP has to be synchronously involved in every exchange, which creates potential bottlenecks[11]. On the other hand, a number of optimistic fair exchange protocols have been proposed in which the TTP participates only if errors occur during the exchange[1, 11, 8, 17]. These protocols are much more efficient and practical for exchanging digital signatures (aka contract signing) or payments and receipts, but they cannot be used to exchange vouchers because they fail to prevent the reproduction of vouchers.

In this paper, we propose a new protocol that enables the fair and effective exchange of electronic vouchers that represent rights. This protocol exchanges vouchers in an optimistic manner; the trading participants first try to exchange vouchers through mutual communication, and they activate the TTP to recover fairness if the exchange is suspended or interrupted and can not be continued. Since this protocol guarantees "strong fairness" in any exchange, fairness can be recovered without identifying or tracking the trading partner.

The rest of the paper is organized as follows: Section 2 states the definitions and requirements for representing vouchers and fairness in voucher trading. This section also describes issues of the legacy method that uses a mediating TTP and previous optimistic fair exchange protocols. Section 3 details the protocol proposed in this paper. Section 4 discusses the proposed protocol by reference to the requirements stated in Section 2.

2. Preliminaries

This section states the definition and requirements of electronic vouchers and fairness in voucher exchanges. In addition, this section discusses drawbacks of the previous approaches.

2.1 Electronic voucher

An electronic voucher is a digital representation of the right to claim services or goods. In this context, the right is created by the issuer, such as a supplier of merchandise or provider of services, as a promise to the right holder. According to RFC3506[5], a voucher is defined as follows:

Electronic voucher Let I be a voucher issuer, H be a voucher holder, and P be the issuer's promise to the right holder. An electronic voucher is defined as the 3-tuple of (I, P, H).

Similar to paper tickets and current money, vouchers should be transferable. The voucher holder H can transfer the voucher (I, P, H) to another participant H'. This transfer is represented as the rewriting of the tuple $(I, P, H) \rightarrow (I, P, H')$. The right lapses from holder H as a result of the transfer.

Vouchers must be protected from illegal acts like forgery, alteration, and reproduction. These security requirements are defined as follows:

Preventing forgery A voucher (*I, P, H*) must be generated (issued) only by issuer *I* and must not be generated by any other participant [1]. In addition, it must not be possible for anyone to alter issuer *I* once the voucher is issued.

Preventing alteration Once a voucher (*I, P, H*) is issued, it must not be possible for anyone to alter promise *P*.

Preventing reproduction It must not be possible for voucher (*I, P, H*) to be reproduced. In particular, in a transfer from *H* to H', both (*I, P, H*) and (I, P, H') must not exist simultaneously throughout the transfer.

2.2 Exchange of vouchers

As mentioned in Section 1, diverse types of commerce transactions can be mapped into exchanges of vouchers. An exchange of vouchers consists of a pair of mutual transfers of vouchers.

To exchange vouchers fairly, it must be assured that no trading participant loses its voucher without receiving the desired voucher; each participant must be able to recover its voucher or the desired voucher in a given period as long as it behaves properly. An exchange of vouchers that satisfies this property is defined as a fair voucher exchange. Details are given below:

Fair voucher exchange Assume that there are two electronic vouchers (I_1, P_1, H_A) and (I_2, P_2, H_B). An exchange constructed of the pair of transfers $(I_1, P_1, H_A) \rightarrow (I_1, P_1, H_B)$ and $(I_2, P_2, H_B) \rightarrow (I_2, P_2, H_A)$ is defined as a fair voucher exchange, if all of the following conditions are satisfied:

1 Provided that H_X (i.e. H_A or H_B) executes the exchange in proper manner, it is assured that H_X will own either voucher as a result of the exchange. That is, either (I_1, P_1, H_X) or (I_2, P_2, H_X) is assured to exist at the termination of the exchange.

2 Provided that H_X executes the exchange in proper manner, it is assured that H_X can terminate the exchange in finite time.

The following conditions must also be satisfied to guarantee the prevention of voucher reproduction:

3 Throughout the exchange, (I_i, P_i, H_A) and (I_i, P_i, H_B) $(i \in \{1, 2\})$ must not exist simultaneously.

[1] Another participant I' may generate voucher (I', P, H) since the issuer is I'.

2.3 Implementation model of vouchers

Although several models can be used to implement electronic vouchers[6], we will focus on the "stored value" model like FlexToken[14]because this type of model has advantages in terms of usability and scalability.

This implementation model assumes that users (owners of vouchers) have tamper-resistant devices like smartcards to store and manage vouchers; it enables vouchers to be transferred among users and redeemed in off-line environments securely without involving banks or other third-parties. In this paper, we refer to these devices as PTDs (personal trusted devices). A PTD protects vouchers from illegal acts like forgery and reproduction from anyone, including its owner.

In this model, a voucher (I, P, H) is considered to exist when user H has PTD U storing digital data v that corresponds to (I, P). In FlexToken, for example, PTDs store 2-tuple entries $(h(PkI), h(R))$, called a token for v, where $h()$ is a secure hash function like SHA-1 [10], PkI is a public key of issuer I, and R is a document called rights definition, which defines the contents of promise P.

The voucher transfer $(I, P, H_A) \rightarrow (I, P, H_B)$ is realized by transferring v from sender's PTD U_A to receiver's PTD U_B, which concludes by deleting v from U_A and storing v in U_B.

2.4 Previous works

While the previous methods of implementing vouchers realize secure circulation of vouchers including issuance, transfer, and redemption with preventing illegal acts like forgery and reproduction, they don't realize fairness in the exchange of vouchers. Although FlexToken discusses fairness in circulating rights, it only aims to ensure the non-repudiation of the fact of circulation; ensuring fairness in voucher exchanges was not considered.

To achieve the fair exchange of vouchers, a voucher trading system that uses a mediating TTP has been introduced[7]. This system satisfies the fairness and security requirements described above, provided that the TTP acts honestly. However, it is weak in terms of availability because every exchange requires synchronous interactions between the TTP and both trading partners; its scalability is also suspect because traffic would be heavily focused on the TTP.

A number of fair exchange protocols for digital signatures and digital data have been proposed[1, 11, 8, 17]. In particular, protocols called optimistic protocols[2, 16, 15]or off-line TTP protocols[18]use a TTP only if errors occur in the exchange process. Under the assumption that errors rarely occur, these protocols relax the problems with using a TTP and enable exchanges to be performed efficiently. These protocols realize, in a practical manner, fair contract signings, certified mails, and fair exchanges of a payment and its receipt

as demonstrated in SEMPER[9, 12], however, no optimistic protocol has been proposed or can be applied for exchanging vouchers.

In order to exchange vouchers fairly without identifying and tracing the trading partner, it is required to ensure fairness without an external dispute resolution process. This level of fairness is called strong fairness[11, 1].

According to [11], strong fairness can be achieved in an optimistic protocol if one of the items to be exchanged ensures "strong generatability" which means the item (or an equivalent item) is generatable by the TTP or "strong revocability" which means the item is revocable by the TTP[2].

We first discuss protocols that use strong generatability and are based upon the protocol introduced in [2]. A voucher is strongly generatable if the TTP could generate a message equivalent to the message that ensures that a voucher is stored while preventing illegal acts on the voucher. However, a voucher is not strongly generatable in this sense because voucher reproduction is possible by replaying the first message of the protocol[3]. This message enables its recipient to perform the resolve protocol with the TTP, which may allow replay of the exchange[4]. This might be harmless for applications like contract signing because it only brings another evidence of confirmation of the same contract concluded in the previous exchange (assuming that contents of the contract are assured to be unique). However, it causes the recipient of the message to reproduce a voucher already received in the previous voucher exchange.

Applying a protocol that uses strong revocability[15]for voucher exchanges is possible but rather impractical. This protocol requires a means by which the TTP can ensure revocation that would prevent the receipt of improperly exchanged vouchers. This is easy for closed-loop electronic money systems (referred to "ECash-like system" in [15]) which involve a bank to confirm payments, but it is difficult for vouchers which have off-line capabilities similar to real tickets or current money; it is impractical to inform revocation to all participants who may receive the revoked voucher before the voucher is transferred or redeemed to them.

We therefore propose a new optimistic protocol for exchanging vouchers that is not based upon either type of protocol mentioned above. Our protocol prevents the replay attacks that make voucher reproduction possible. In addition, the protocol is simple and efficient; its main protocol consists of four messages, two of which are signed, and is as simple as a naive voucher exchange involving the mutual transfer of two vouchers using challenge and response.

[2]The other item is required to be "weakly generatable", which is rather easy to achieve.
[3]Another replay attack against this protocol is pointed out in [13], but it can be fixed easily as described in that paper.
[4]Additional message exchanges could prevent this replay attack, but it would make the whole protocol much more complicated.

3. Fair Exchange Protocol for Vouchers

This section describes a fair exchange protocol that satisfies the requirements stated in Section 2.

In this section, we assume that the vouchers to be exchanged are (I_1, P_1, H_A) and (I_2, P_2, H_B), the implementation model is stored value model, (I_i, P_i) is represented by token v_i, and voucher (I_i, P_i, H_X) exists when v_i is stored in **PTD** U_X held by user H_X. The set of tokens stored in U_X is referred to as V_X. **PTD** U_X is a tamper-resistant device like a smartcard which is capable of preventing illegal alteration of V_X or the program performing the exchange, as well as keeping its signing key and n_2 (described in 3.2) secret. Each process in the PTD is assumed to be performed atomically.

User H_X would also have a user terminal device such as a mobile phone or a PDA to interact with its **PTD** U_X if the PTD doesn't have a user interface (like a smartcard). The user terminal would facilitate the generation of communication channels among PTDs and TTP *T* (see below) as well, but it doesn't have to be trusted by anyone but its owner; the owner might try to cheat its PTD or another PTD by forging or replaying messages using the terminal. The terminals aren't shown explicitly hereafter since they can be merely treated as a part of (insecure) communication channels in the proposed protocol.

This protocol consists of three sub-protocols: a main protocol, an abort, and a resolve protocol. An exchange starts by performing the main protocol. The exchange completes only within the main protocol whenever both participants act honestly and there is no trouble in the communication channel between them. If there are any troubles in the main protocol, either participant can recover fairness and terminate the exchange with the abort protocol or the resolve protocol using TTP *T,* which is a third-party trusted by both participants (and also the issuers of the vouchers). This protocol exchanges vouchers fairly and optimistically.

3.1 Definitions

The other definitions and assumptions needed to describe the proposed protocol are given below (details are given in Section 4).

U_A has public key certificate $Cert_A$ including public key PkA, a signing key which generates a signed message $(m)_{PkA}$ verifiable with $Cert_A$, and a verify function $Verify((m)_{PkX}, Cert_X)$ which verifies the signed message $(m)_{PkX}$ using the corresponding certificate $Cert_X$; and U_B and *T* likewise.

$Cert_A$ and $Cert_B$ represent that their keyholders U_A and U_B are certified as proper PTDs, while $Cert_T$ represents that its keyholder *T* is certified as a proper TTP. Note that these certificates don't have to be identity certificates to identify individuals, but have to be issued by a party who is trusted by the issuer of the vouchers being exchanged. It is easy to ensure this requirement if the

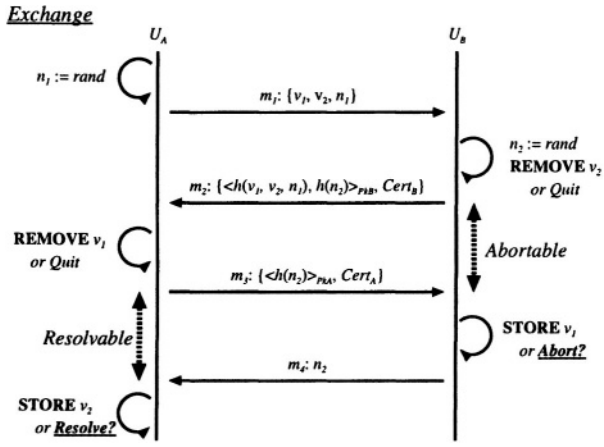

Figure 1. Main protocol.

existence of a certificate authority trusted by all participants can be assumed. If not, additional certificates by the issuer like accredit information in FlexToken can be used (see details in [14]).

In addition, U_A and U_B manage sets of sessions S_A and S_B respectively, and T manages sets S_{abort} and $S_{resolve}$ which include sessions that have been aborted or resolved.

3.2 Main protocol

Figure 1 shows the main protocol. At the initial point, tokens v_1 and v_2 are stored in U_A and U_B respectively ($v_1 \in V_A, v_2 \in V_B$). If this protocol terminates without failure, v_1 and v_2 are swapped ($v_1 \in V_B, v_2 \in V_A$). The exchange is performed without intervention of the TTP in this case. In the explanation hereafter, it is assumed that H_A knows the content of v_2 at the initial point for simplicity.

H_B may order U_B to abandon this protocol at any time in the abortable section in Figure 1 (after sending m_2 and before receiving m_3) and terminate the exchange by performing the abort protocol with T which recovers fairness for H_B. Likewise, H_A may order U_A to terminate the exchange by performing the resolve protocol with T throughout the resolvable section (after sending m_3 and before receiving m_4). Before these sections, H_A and H_B can merely quit the exchange without losing fairness.

H_A starts the main protocol by ordering U_A to exchange v_1 in U_A and v_2 in U_B. The main protocol is performed in the following way:

1 After receiving the order from H_A, U_A performs the following:

(a) Generates a random number n_1 and adds it to set S_A.

(b) Sends $m_1 : \{v_1, v_2, n_1\}$ to U_B, which is the offer of the exchange.

2 U_B receives m_1 and performs the following:

(a) Confirms m_1 if it is an acceptable offer for H_B. If not, U_B quits the exchange (and may inform U_A of this event).

(b) Generates n_2 and adds it to set S_B; n_2 has to be kept in secret until it is sent as m_4.

(c) Removes v_2 from V_B, which causes the corresponding rights to lapse from H_B; H_B temporarily falls into unfair condition.

(d) Calculates $s_1 := h(v_1|v_2|n_1)$ and $s_2 := h(n_2)$.

(e) Sends $m_2 : \{(s_1|s_2)_{PkB}, Cert_B\}$ to U_A.

3 U_A receives m_2 and performs the following:

(a) Confirms that all of the following equations are satisfied.

i $s_1 = h(v_1|v_2|n_1)$

ii $Verify(m_2) = true$

If not, U_A waits m_2 again or quits the exchange by removing n_1 from S_A.

(b) Removes n_1 from S_A and adds s_2 to S_A.

(c) Removes v_1 from V_A, which causes the corresponding rights to lapses from H_A; H_A temporarily falls into unfair condition (That is, H_A and H_B fall into unfair condition at this time).

(d) Sends $m_3 : \{(s_2)_{PkA}, Cert_A\}$ to U_B.

4 U_B receives m_3 and performs the following:

(a) Confirms that all of the following equations are satisfied.

i $s_2 = h(n_2)$

ii $Verify(m_3) = true$

If not, U_B waits m_3 again or abandons the main protocol and performs the abort protocol to recover fairness.

(b) Removes n_2 from S_B and adds v_1 included in m_1 to V_B. H_B enters fair condition again.

(c) Sends $m_4 : n_2$ to U_A, and U_B terminates the exchange.

5 U_A receives m_4 and performs the following:

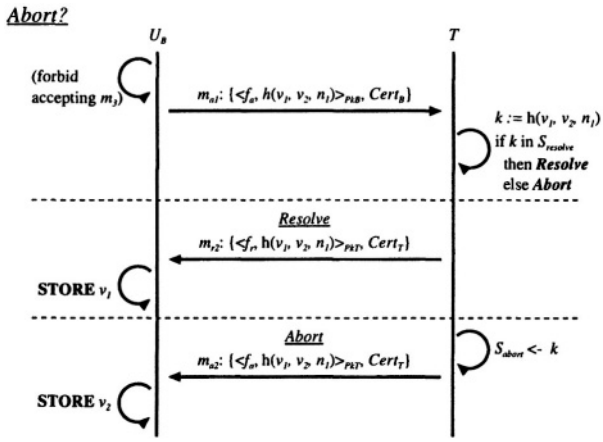

Figure 2. Abort protocol.

(a) Confirms $h(m_4) = s_2$. If not, U_A waits m_4 again or performs the resolve protocol to recover fairness.

(b) Removes s_2 from S_A and adds v_2 to V_A. H_A enters fair condition again, and U_A terminates the exchange.

3.3 Abort protocol

Figure 2 shows the flow of the abort protocol. As mentioned above, U_B may abandon the main protocol at any time in the abortable section by performing this protocol.

The abort protocol enables U_B to recover fairness by sending an abort request m_{a1} to T and receiving the abort admission m_{a2} which allows U_B to restore v_2, or the resolve admission m_{r2} to store v_1 if T has already received the resolve request from U_A. If U_B cannot receive either m_{a2} or m_{r2} from T in a given period, U_B may resend m_{a1}.

The abort protocol is performed as follows:

1 U_B abandons the main protocol, and is prohibited from receiving m_3 in the main protocol.

2 U_B sends $m_{a1} : \{(f_a|s_1)_{PkB}, Cert_B\}$ to T. Herein, f_a is a flag which represents the process of aborting.

3 T receives m_{a1} and performs the following:

(a) Confirms $Verify(m_{a1})$. If not, waits m_{a1} or m_{r1} (described in the resolve protocol) again.

(b) Let $k := s_1$,

i If $k \in S_{resolve}$, then send the resolve admission.

ii If $k \notin S_{resolve}$, then add k to S_{abort} and send the abort admission.

The procedures U_B performs when U_B receives the abort admission or the resolve admission are as follows:

Receiving abort admission

1 T sends $m_{a2} : \{(f_a|k)_{PkT}, Cert_T\}$ to U_B.

2 U_B receives m_{a2} and performs the following:

(a) Confirms $k = s_1$ and $Verify(m_{a2})$. If not, waits m_{a2} or m_{r2} again.

(b) Removes n_2 from S_B.

(c) Adds v_2 to V_B and terminates the exchange.

Receiving the resolve admission

1 T sends $m_{r2} : \{(f_r|k)_{PkT}, Cert_T\}$ to U_B.

2 U_B receives m_{r2} and performs the following:

(a) Confirms $k = s_1$ and $Verify(m_{r2})$. If not, waits m_{a2} or m_{r2} again.

(b) Removes n_2 from S_B.

(c) Adds v_1 to V_B and terminates the exchange.

3.4 Resolve protocol

Figure 3 shows the flow of the resolve protocol. U_A may perform this protocol at any time in the resolvable section of the main protocol.

Similar to the abort protocol, the resolve protocol enables U_A to recover fairness by sending a resolve request m_{r1} to T and receiving the abort admission m_{a2} which allows U_A to restore v_1, or the resolve admission m_{r2} to store v_2 if T has already received the abort request from U_B. If U_A cannot receive either m_{a2} or m_{r2} from T in a given period, U_A may resend m_{r1}.

The resolve protocol is performed as follows:

1 U_A sends $m_{r1} : \{(f_r|s_1)_{PkA}, Cert_A\}$ to T. Herein, f_r is a flag that represents the process of resolving.

2 T receives m_{r1} and performs the following:

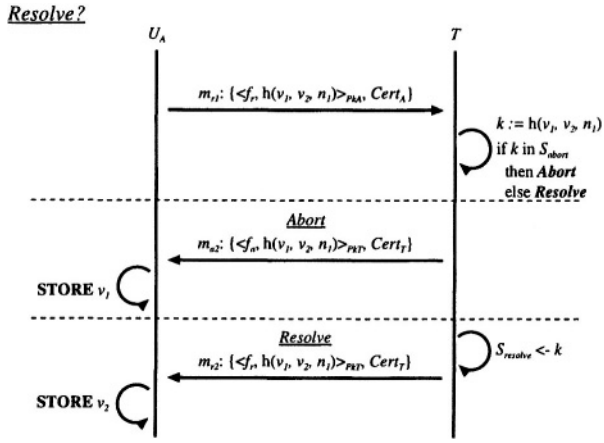

Figure 3. Resolve protocol.

(a) Confirms $Verify(m_{r1})$. If not, waits m_{a1} or m_{r1} again.

(b) Let $k := s_1$,

i If $k \in S_{abort}$, then send the abort admission,

ii If $k \notin S_{abort}$, then add k to $S_{resolve}$ and send the resolve admission.

U_A processes the abort admission or the resolve admission in a similar way to U_B in the abort protocol, except for the following differences:

- U_B is replaced by U_A.
- Deleting n_2 from S_B is replaced by deleting s_2 from S_A.
- v_1 and v_2 stored by receiving m_{r2} or m_{a2} are swapped.
- U_A doesn't have to abandon the main protocol. U_A may conclude the exchange by receiving m_4 in the main protocol before receiving m_{a2} or m_{r2}, because it can be assumed that U_B has successfully concluded the exchange when U_A receives m_4 and therefore there is no chance for U_A to receive the abort admission m_{a2}.

4. Discussions about Security and Fairness

This section discusses how the proposed protocol ensures the security requirements for vouchers and the fairness requirement.

In the discussion hereafter, the following assumptions are made:

1 Users H_A and H_B may try to cheat their PTDs; they might replay or forge any messages.

2 Tamper-resistant capability of both PTDs, U_A and U_B, is not compromised; they process the input data m_i properly and their signing keys are kept secret. In addition, n_2 is kept secret until it is sent as m_4.

3 TTP T properly processes abort requests and resolve requests in a given period.

4 The communication channels linking U_A, U_B and T may be insecure, i.e. attacks including eavesdropping, replaying and alteration are possible.

5 The communication channel between U_A and U_B may be lost permanently (i.e. either partner might escape in the middle of the exchange and the other partner is not assured of catching him).

6 The communication channel between T and U_A or U_B may be lost, but it recovers in a given period.

7 The hash function and signatures used in the protocol are sufficiently secure; e.g. the risk of collision of the hash function or forgery of the signatures can be ignored.

8 The certificate practice is secure enough; e.g. no certificate involved in the protocol can be forged or improperly issued.

4.1 Preventing forgery and alteration

In the proposed protocol, forgery or alteration is possible if it is possible to store a token v' different from either v_1 or v_2 included in offer message m_1.

There is no chance to store a different $v'(\notin \{v_1, v_2\})$ to U_A because the offer m_1 originates from U_A itself. H_A, H_B or another faulty party may try to store $v_i(\notin \{v_1, v_2\})$ to U_B by altering v_1 or v_2 included in m_1, but this would be detected by its inconsistent hash value at step 3 in the main protocol.

Assuming v_1 in m_1 is altered to v', s_1 included in the m_2 (protected by signature of U_B) becomes $s_1 = h(v', v_2, n_1)$, which is inconsistent with $h(v_1, v_2, n_1)$ as indicated by the comparison in step 3 of the main protocol.

The risk of forgery and alteration therefore depends on the collision resistance of hash function $h()$ and the strength of the signature, which are assumed to be sufficient.

Forgery and alteration are, therefore, prevented in the proposed protocol.

4.2 Preventing reproduction

In order to prevent reproduction, there must not be any token storing operation without the preceding corresponding removal operation of the same token.

In the main protocol, any storing operation of token v_i is performed after removing v_i. In addition, both m_2 and m_3, the evidence of the removal of v_2 and v_1, respectively, are protected against forgery or alteration by the attached signatures, and they cannot be replayed because storing v_1 and v_2 always involves the removal of n_2 and s_2 respectively, which prevents the overlapped storing of tokens. The main protocol thus prevents the reproduction of vouchers.

A discussion about abusing the recovery protocols is more complex. We discuss the possibility of reproducing v_1 and v_2 separately.

Preventing reproduction of v_1. v_1 is reproduced when both of the following are performed:

1 U_A stores v_1 in the resolve protocol.

2 U_B stores v_1 in the main protocol or the abort protocol.

In order for U_A to store v_1 in the resolve protocol, U_A needs to receive abort admission m_{a2}, which is signed and sent from T only if T has received abort request m_{a1} from U_B before receiving resolve request m_{r1} from U_A.

Assuming U_B sent m_{a1}, U_B cannot store v_1 in the main protocol because the main protocol is abandoned when U_B starts the abort protocol. Neither can U_B store v_1 in the abort protocol because U_B needs to receive resolve admission m_{a2}, which is sent from T only if T received m_{r1} before receiving m_{a1}. Since this contradicts the above condition for U_A to store v_1, v_1 cannot be reproduced by abusing the recovery protocols.

Preventing reproduction of v_2. v_2 is reproduced when both of the following are performed:

1 U_A stores v_2 in the main protocol or the resolve protocol.

2 U_B stores v_2 in the abort protocol.

For U_A to store v_2 in the main protocol, U_A needs to receive m_4, which is only known by U_B and cannot be guessed by any other participant. If U_B sent m_4, U_B cannot store v_2 because U_B ought to have stored v_1 and already terminated the exchange before sending m_4.

If U_A is to store v_2 in the resolve protocol, U_A needs to receive resolve admission m_{r2}, which is signed and sent from T only if T has *not* received abort request m_{a1} before receiving resolve request m_{r1}. Meanwhile, in order to let U_B store v_2 in the abort protocol, U_B needs to receive abort admission m_{a2}, which is signed and sent from T only if T has *not* received resolve request m_{r1} before receiving abort request m_{a1}. Since these two conditions contradict each other, v_2 cannot be reproduced by abusing the recovery protocols.

4.3 Ensuring fairness

In order to ensure fairness in voucher exchange, the following must be satisfied according to the definition of fairness described in Section 2:

1 Both U_A and U_B can obtain either token v_1 or v_2 (in the set V_A and V_B) when the exchange is terminated.

2 Both U_A and U_B can terminate the exchange in finite time.

3 v_1 must not be obtained by both U_A and U_B simultaneously throughout the exchange; v_2 likewise.

The last condition is derived from the requirement to prevent reproduction as already discussed in Section 4.2. We therefore discuss the remaining two conditions hereafter.

In the main protocol, U_A falls into unfair condition in which it has neither v_1 nor v_2 in V_A while in the resolvable section, and U_B falls into unfair condition while in the abortable section. Except in these sections, they have either v_1 or v_2 and can instantly quit the exchange at will.

While in the resolvable section, U_A is assured of recovering fairness and can terminate the exchange in a given period, because U_A may perform the resolve protocol at any time in the resolvable section, and the resolve protocol concludes in a given period provided that assumptions 3 and 6 described at the beginning of this section are satisfied. U_B in the abortable section is also assured of recovering fairness in a similar manner.

The proposed protocol therefore guarantees fairness for both U_A and U_B.

5. Conclusion

In this paper, we stated the definitions and requirements for vouchers and fairness in voucher exchanges as a basis of realizing reliable electronic commerce, and proposed a new practical protocol that enables fair and efficient (optimistic) exchanges of vouchers stored in trusted devices like smartcards, while preventing illegal acts on vouchers such as reproduction and forgery.

This protocol enables the trading participants to exchange vouchers without recourse to a TTP provided there is no accident in the exchange; it guarantees the recovery of fairness and safe termination of the exchange in a given period by accessing a TTP even if there are problems with the communication channel or dishonest acts by the partner. These properties eliminate the problems inherent in the previous method that uses a mediating TTP, including the scalability problem (traffic concentrated on the TTP) and the availability problem (TTP involved in each trade).

This protocol is thus practical and enables people participating in electronic commerce to trade with unidentified partners in complete confidence.

References

[1] N. Asokan. *Fairness in Electronic Commerce.* PhD thesis, University of Waterloo, 1998.

[2] N. Asokan, V. Shoup, and M. Waidner. Asynchronous protocols for optimistic fair exchange. In *Proceedings of the IEEE Symposium on Research in Security and Privacy,* pages 86–99. IEEE Computer Society Press, May 1998.

[3] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications,* 18(4):593–610, Apr. 2000.

[4] C. Ellison and B. Schneier. Ten risks of PKI: What you're not being told about public key infrastructure. *Computer Security Journal,* 16(1): 1–7, 2000.

[5] K. Fujimura and D. Eastlake. *RFC 3506: Requirements and Design for Voucher Trading System (VTS),* Mar. 2003.

[6] K. Fujimura and M. Terada. Trading among untrusted partners via voucher trading system. In *Proceedings of the 1st Conf. on e-Commerce, e-Business, e-Government.* IFIP, Oct. 2001.

[7] M. Iguchi, M. Terada, Y. Nakamura, and K. Fujimura. Voucher-integrated trading model for C2B and C2C e-commerce system development. In *Proceedings of the 2nd Conf. on e-Commerce, e-Business, e-Government.* IFIP, Oct. 2002.

[8] S. Kremer, O. Markowitch, and J. Zhou. An intensive survey of fair non-repudiation protocols. *Computer Communications,* 25(17):1606–1621, Nov. 2002.

[9] G. Lacoste, B. Pfitzmann, M. Steiner, and M. Waidner, editors. *SEMPER — Secure Electronic Marketplace for Europe,* volume 1854 of *LNCS.* Springer-Verlag, New York, NY, USA, 2000.

[10] NIST. *F1PS 180-1: Secure Hash Standard,* Apr. 1995.

[11] H. Pagnia, H. Vogt, and F. C.Gärtner. Fair exchange. *The Computer Journal,* 46(l):55–75, Jan. 2003.

[12] M. Schunter. *Optimistic Fair Exchange.* PhD thesis, Universität des Saarlandes, 2000.

[13] V. Shmatikov and J. C. Mitchell. Analysis of a fair exchange protocol. In *Proceedings of the 1999 FLoC Workshop on Formal Methods and Security Protocols,* July 1999.

[14] M. Terada, H. Kuno, M. Hanadate, and K. Fujimura. Copy prevention scheme for rights trading infrastructure. In *Proceedings of the 4th Working Conference on Smart Card Research and Advanced Applications (CARDIS),* pages 51–70. IFIP, Sept. 2000.

[15] H. Vogt. Asynchronous optimistic fair exchange based on revocable items. In *Proceedings of the 7th International Financial Cryptography Conference,* pages 208–222. IFCA, Jan. 2003.

[16] H. Vogt, H. Pagnia, and F. C. Gärtner. Modular fair exchange protocols for electronic commerce. In *Proceedings of the 15th Annual Computer Security Applications Conference,* pages 3–11, Dec. 1999.

[17] J. Zhou. Achieving fair nonrepudiation in electronic transactions. *Journal of Organizational Computing and Electronic Commerce,* 11(4):253–267, Dec. 2001.

[18] J. Zhou, R. Deng, and F. Bao. Evolution of fair non-repudiation with TTP. In *Proceedings of the 4th Australasian Conference on Information Security and Privacy (ACISP'99),* pages 258–269, Apr. 1999.