Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

ELSEVIER

Computer Networks 41 (2003) 19–28

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

An efficient protocol for anonymous and fair document exchange

N. Zhang [a], Q. Shi [b,*], M. Merabti [b]

[a] *Department of Computer Science, The University of Manchester, Oxford Road, Manchester M13 9PL, UK*
[b] *School of Computing & Mathematical Sciences, Liverpool John Moores University, Byrom Street, Liverpool L3 3AF, UK*

Received 31 March 2001; received in revised form 18 March 2002; accepted 10 June 2002

Responsible Editor: P. Dowd

Abstract

Fairness in document exchange has been well studied, while anonymity in the exchange, which protects the privacy of personal information such as identities and locations, has been either ignored or handled with partial or inappropriate considerations. In this paper we propose a new protocol for anonymous and fair document exchange between two parties with the assistance of an off-line trusted third party. The new protocol treats both fairness and anonymity as essential properties, employs an efficient method for off-line key recovery, and places weak requirements on the security of the third party.

Keywords: Electronic commerce; Fair document exchange; Anonymity; Communication protocol

1. Introduction

Exchange of valuable documents between two parties (e.g. companies, organisations or individuals) is an important activity of electronic commerce, and its applications include exchange of valuable information, and exchange of valuable electronic goods for a payment. Due to the valuable nature (e.g. payments) of the documents, the exchange must be fair and secure to avoid the situation where one party can receive its expected document, while the other cannot. Another important property is anonymity that protects the privacy of personal information. For instance, an individual, who engages in an exchange with a vendor, would like to conceal his/her identity to prevent the vendor from assembling the profile of his/her personal interests, life-styles, whereabouts, etc.

So far a number of protocols have been proposed to achieve fair exchange [1–6,8,10–14,16]. The main approach used by the protocols is based on a trusted third party that acts as an intermediary to assist in document exchange. The role of the trusted party can be divided into on-line and off-line. An on-line trusted third party actually

* Corresponding author. Tel.: +44-151-2312272; fax: +44-151-2074594.
E-mail addresses: nzhang@cs.man.ac.uk (N. Zhang), q.shi@livjm.ac.uk (Q. Shi).

1389-1286/03/$ - see front matter
PII: S1389-1286(02)00323-7

takes part in an exchange process, e.g. collecting, verifying and forwarding data items related to keys for document decryption [11]. An off-line trusted third party does not participate in the exchange process in normal cases, and is only invoked to help to complete the exchange in abnormal cases where the exchange is not operated properly due to system faults or a party's misbehaviour [1–6,8]. Fair exchange protocols based on an off-line trusted third party are preferable as they offer a more cost-effective use of a trusted third party.

However, the existing fair exchange protocols either do not consider anonymity [5] or have partial or inappropriate considerations of anonymity [3,11,14]. This has motivated us to propose a new protocol for anonymous and fair document exchange between two parties with the assistance of an off-line trusted third party. The main contributions of our protocol are threefold. First, it offers not only good fairness but also true anonymity. Secondly, it provides a simpler and more efficient off-line recovery method for handling abnormal cases of exchange than other existing methods. Thirdly, it places weak requirements on the security of the trusted third party to simplify the management and protection of the party.

The rest of the paper is organised as follows. Section 2 summarises the notation used by the new protocol, and Section 3 states the assumptions employed for the protocol design. Based on the notation and assumptions, we define the protocol in Section 4, and present a method for off-line recovery in Section 5. The fairness and anonymity of the new protocol are examined in Section 6. Finally our conclusions are outlined in Section 7.

2. Notation

The notation to be used throughout this paper is summarised as follows:

- $E_k(x)$ expresses the ciphertext of a data item x encrypted with a key k. $E_k(x)$ is computed using a public-key cryptosystem if the corresponding decryption key is not k, and using a conventional cryptosystem otherwise.
- $f(x) = x^2 \bmod n$ is a one-way function where *mod* denotes the modulo operator, n is a product of two large distinct primes, and each of the domain and range of $f()$ is Z_n^* (i.e. the set of all positive integers less than and relatively prime to n) [11]. $f()$ is one-way under the assumption that n is hard to factor.
- $h(x)$ is a one-way hash function with the following properties: (a) for any x, it is easy to compute $h(x)$; (b) given $h(x)$, it is hard to compute x; and (c) given x, it is hard to find $x'(\neq x)$ such that $h(x) = h(x')$.
- x, y denotes the concatenation of data items x and y.
- $P_a \rightarrow^A P_b$: m signifies that a party P_a sends a message m to another party P_b over an anonymous communication channel [7]. Detailed discussion on such communication will be given in Section 4.
- co_b is a party P_b's commitment to a document exchange. It assures another party P_a that a designated trusted third party can help P_a to recover a required document decryption key k_b of P_b from co_b, and P_a can verify the correctness of such assurance without knowing k_b. The formation and verification of co_b will be presented in detail in Section 5.

3. Assumptions

Suppose that a party P_a has a valuable document D_a and a conventional (or symmetric) key k_a for the encryption and decryption of D_a. Similarly another party P_b has a valuable document D_b and a conventional key k_b. P_a and P_b wish to anonymously and fairly exchange their documents D_a and D_b, and have agreed to employ a party P_t as an off-line trusted third party in assistance with the exchange process.

Before the exchange starts, we require these parties to meet the following assumptions:

(1) P_a knows $hd_b = h(E_{k_b}(D_b))$ (i.e. the hash value of the ciphertext of D_b encrypted with k_b) and $f(k_b)$ where the correctness of D_b and k_b has been certified or confirmed by an authority. P_b knows $hd_a = h(E_{k_a}(D_a))$ and $f(k_a)$ where

the correctness of D_a and k_a has also been certified by an authority.

(2) P_a has a pair of public and private keys, pk_a and sk_a, and similarly P_b has a pair of public and private keys, pk_b and sk_b. Each party knows the other party's public key.

(3) P_t has a certificate for its public key pk_t, which is already held by both P_a and P_b. The public-key cryptosystem used by P_t is based on the RSA algorithm [15], i.e. P_t's public and private keys can be denoted as $pk_t = \{e_t, n_t\}$ and $sk_t = \{d_t, n_t\}$ with n_t being a product of two distinct large primes.

(4) n in the one-way function $f()$ defined in Section 2 is equal to n_t, i.e. $n = n_t$. This does not affect the one-way property of $f()$, as n_t should be a product of two large (100–200 digits or even larger) distinct primes and hard to factor.

Assumption (1) above deserves more explanation. In fact, it is similar to the assumption used and justified in [11]. This assumption is essential for achieving the fairness of the exchange, as it allows each party to verify the correctness of the other's encrypted document and decryption key during the exchange. Without this assumption, a dishonest party could use a worthless document to exchange for the valuable document with the other party, and this dishonesty is not detectable until the exchange is completed. In other words, when the dishonesty is detected, the dishonest party has already unfairly gained the other party's valuable document.

hd_b and $f(k_b)$ in assumption (1) can be certified or confirmed together with public key pk_b in assumption (2). As shown in [11], the certification may take the form $\{desc_b, hd_b, f(k_b), pk_b, sign_b\}$ where $desc_b$ is a description of the contents of document D_b (e.g., if D_b is a movie, then $desc_b$ is the title and summary of the movie), and $sign_b$ is the certification authority's signature on items $desc_b$, hd_b, $f(k_b)$ and pk_b. The purpose for the inclusion of pk_b is to allow another party to send messages to P_b securely. Note that if P_b wishes to remain anonymous, its public key pk_b should not be bound to its identity, and different documents of P_b could use different public keys. The signature $sign_b$ represents the authority's approval that if an encrypted document ed_b and a key k'_b meet the condition $h(ed_b) = hd_b$ and $f(k'_b) = f(k_b)$, then the decryption of ed_b with k'_b will recover a document (i.e. D_b) with its contents matching the description in $desc_b$. Note that the authority only needs to issue the certificate $\{desc_b, hd_b, f(k_b), pk_b, sign_b\}$ once, which is then used by P_b to exchange D_b for as many other documents as P_b can.

Similar certification is also applied to hd_a, $f(k_a)$ and pk_a.

To illustrate the application of the above document certification, consider an example of on-line digital goods purchases in e-commerce. Let D_b represent a film produced and certified by a film producer, and P_b an on-line merchant contracted by the film producer to sell the film on-line. The film producer only needs to certify the film D_b for P_b once, i.e. it issues $\{desc_b, hd_b, f(k_b), pk_b, sign_b\}$. Here, P_b could determine keys k_b and pk_b, and securely pass them to the film producer (e.g. the keys can be encrypted with the producer's public key) for the certification. P_b can then sell D_b for as many times as P_b can without any involvement of the film producer. Obviously, in this case, P_b does not want to be anonymous, and would like to publicise the film as much as possible to boost its sale.

A customer denoted as P_a wants to purchase a copy of the film after seeing an advertisement about it on TV, but for the sake of privacy, P_a does not like to disclose his/her identity and location to the vendor P_b. To purchase the film anonymously, P_a obtains an anonymous electronic payment (e.g. electronic cash) expressed as D_a, which is certified or issued as $\{desc_a, hd_a, f(k_a), pk_a, sign_a\}$ by a major credit card company or bank. P_a wants to fairly and anonymously exchange D_a for D_b with P_b.

In this example, the major credit card company or bank and the film producer play the role of the authorities for the certification of documents D_a and D_b, respectively. This indicates that an authority may not have to certify both documents to be exchanged, and the document certification may be an inherent process rather than a separate process, e.g. payment D_a does not need certification from another authority. Additionally, an authority could be off-line, e.g. P_b receives a CD of the film from the film producer by post and then uploads the film from the CD to a server.

4. Protocol for anonymous and fair document exchange

Before presenting the protocol, we first discuss the issue of anonymity involved in document exchange. Anonymity can be divided into three cases. In the first case, one party wishes to remain anonymous, while the other does not. For instance, a vendor selling electronic goods on the Internet would not like the true identity to be concealed as this may affect customers' confidence in the vendor. On the other hand, a customer would like to be anonymous in purchasing goods from the vendor so as to protect the privacy of his/her identity and location.

In the second case, both parties wish to remain anonymous. For example, an individual P_a, who wants to anonymously exchange a valuable file for another one, can advertise it on a Web site or in a news group by providing an anonymous contact address. In response to P_a's advertisement, another individual P_b, who has the file wanted by P_a and likes to anonymously exchange it for P_a's file, can contact P_a using the address provided and reach an agreement on the exchange.

In the third case, both parties know each other, but do not want any other party to know that they are involved in an exchange. For example, when a company negotiates with another company on a possible takeover, the two companies may not like any other party to know this fact by observing a series of exchanges between them, before the negotiation is completed.

The protocol to be presented below does not distinguish the three cases above. In fact, the main difference lies in anonymous communication channels. For the first case, the anonymous party defines its anonymous channel(s) for sending a message to the other party and allowing the other party to respond to its message. For the second case, when a party P_b wants to send a message to the other party P_a, P_b first defines an anonymous channel and then connects it to the anonymous channel specified by P_a's contact address. This can protect the anonymity of both parties. For the third case, an anonymous channel is defined by each party to communicate with the other to prevent any other party from using the messages transferred between the two parties to link them together.

One of the principal ideas used to establish an anonymous communication channel is based on a series of nodes, called *mixes*, through which a message is transferred from its sender to its recipient [7]. Each mix may collect a number of messages usually with a constant size achieved by padding random data if necessary, change their outlooks by a cryptographic operation, and send them out in a different order. This makes it very difficult for a traffic observer to follow up the passage of a message through the mixes. This approach allows a message sender to anonymously transmit a message possibly with an anonymous reply address to enable a recipient to respond. For further information on anonymous communication, please refer to [7].

Although anonymous communication can conceal the location and identity of a message sender from a message recipient and a traffic observer, such communication itself is insufficient to achieve anonymous document exchange. This is because the contents of messages exchanged may reveal the identity of a party, and the information used by P_t to handle an abnormal case of exchange may unnecessarily disclose the identity of a party to P_t, as seen in existing fair exchange protocols [5]. We therefore need not only anonymous communication but also a document exchange protocol built on it, to fulfil anonymous document exchange. Existing work does not offer such a protocol appropriately.

We can now describe how parties P_a and P_b anonymously and fairly exchange their documents with the assistance of trusted third party P_t. This exchange process needs to meet the following fairness and anonymity requirements:

(a) Exchange fairness: At the end of the exchange, if P_a has obtained P_b's document D_b or can obtain D_b with the assistance of P_t, then P_b has obtained P_a's document D_a or can obtain D_a with the assistance of P_t, and vice versa.
(b) Exchange anonymity: During the exchange, P_a cannot use messages received from P_b to infer the identity and location of P_b if P_b wishes to remain anonymous, and vice versa. Additionally,

P_t has no need to know the identities and locations of P_a and P_b as well as the contents of the documents exchanged, when P_t is invoked to handle an abnormal case of exchange.

These requirements ensure that either each of P_a and P_b or neither of them can get the other's document, and that each of P_a and P_b can conceal its identity and location from the other party and P_t if it wishes to remain anonymous. The purpose for hiding the identities and locations of P_a and P_b from P_t is to weaken security requirements on P_t so as to simplify the implementation and management of P_t.

A document exchange process satisfying the above requirements is presented below, which comprises the following steps:

1. P_a (anonymously) sends to P_b its encrypted document $E_{k_a}(D_a)$ and an item kr_a for the computation of key k_a at step 3.
2. P_b sends to P_a its encrypted document $E_{k_b}(D_b)$ and an item kr_b for the computation of k_b at step 4 together with P_b's commitment co_b produced in relation to kr_a and kr_b, if P_b has successfully verified $E_{k_a}(D_a)$ received. co_b assures P_a that P_t can help P_a recover k_b from co_b, and P_a can verify the correctness of such assurance without knowing k_b.
3. P_a sends an item r_a to P_b if P_a has successfully verified $E_{k_b}(D_b)$ and co_b, and P_b then uses r_a to compute k_a from kr_a.
4. P_b sends an item r_b to P_a if key k_a obtained is correct, and P_a then uses r_b to compute k_b from kr_b.
5. P_a invokes P_t to recover r_b from co_b for the computation of k_b from kr_b, only if P_a has failed to receive r_b from P_b at step 4 after having sent r_a to P_b at step 3.

This exchange process will be formalised into a protocol consisting of two sub-protocols. The first sub-protocol corresponds to the first four steps of the above exchange process, namely normal cases of exchange without the key recovery. The other sub-protocol is for the last step, i.e. the key recovery in case a normal exchange has failed.

We begin with the first sub-protocol that is defined in Table 1 including the definitions of all the items used. For intuitiveness, the sub-protocol is also shown as the Unified Modeling Language (UML) sequence diagram [9] in Fig. 1. The transactions of the sub-protocol are explained below:

Table 1
Document exchange sub-protocol

Sub-protocol 1	
E_1. $P_a \rightarrow^A P_b$: $E_{k_a}(D_a)$, $E_{pk_b}(sn, kr_a, aca_{a,1})$	
E_2. $P_b \rightarrow^A P_a$: sn, $E_{k_b}(D_b)$, $E_{pk_a}(kr_b, co_b)$	
E_3. $P_a \rightarrow^A P_b$: sn, r_a	
E_4. $P_b \rightarrow^A P_a$: sn, r_b	
Item	Definition
sn	Session number
k_a, D_a	P_a's symmetric key and document, respectively
pk_b, sk_b	P_b's public and private keys, respectively
pk_t, sk_t	P_t's RSA public and private keys, i.e. $pk_t = \{e_t, n_t\}$ and $sk_t = \{d_t, n_t\}$
r_a	Random number picked by P_a
kr_a	$= (k_a \times r_a^{-1}) \bmod n_t$, computed by P_a
$aca_{a,1}$	P_a's anonymous contact address
k_b, D_b	P_b's symmetric key and document, respectively
pk_a, sk_a	P_a's public and private keys, respectively
r_b	Random number picked by P_b
kr_b	$= (k_b \times r_b^{-1}) \bmod n_t$, computed by P_b
co_b	P_b's commitment

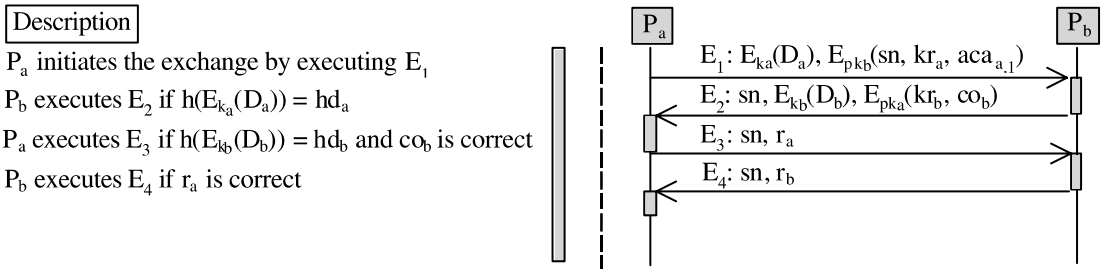

Fig. 1. Document exchange sub-protocol.

E_1: In this transaction (i.e. step 1), P_a chooses a random number r_a from the domain of $f()$, and computes:

$$kr_a = (k_a \times r_a^{-1}) \bmod n_t,$$

where $r_a{}^{-1}$ is the multiplicative inverse of r_a, i.e. $(r_a \times r_a^{-1}) \bmod n_t = 1$. sn in E_1 is a session number (chosen by P_a, or agreed by both P_a and P_b before the exchange begins) to distinguish the current session from previous ones. $aca_{a,1}$ is P_a's anonymous contact address to allow P_b to reply to the message. If P_a does not want to be anonymous, $aca_{a,1}$ can be its normal contact address. Here we assume that P_a knows P_b's (anonymous) contact address. P_a encrypts sn, kr_a and $aca_{a,1}$ with P_b's public key pk_b, i.e. $E_{pk_b}(sn, kr_a, aca_{a,1})$. P_a then transfers its message to P_b over an anonymous channel.

Upon reception of the message, P_b decrypts $E_{pk_b}(sn, kr_a, aca_{a,1})$ with private key sk_b to obtain sn, kr_a and $aca_{a,1}$ If $h(E_{k_a}(D_a))$ is equal to hd_a possessed (see the assumption in Section 3), then P_b performs transaction E_2 and otherwise P_b requires P_a to retransmit the message. Note that it is hard for P_b to compute k_a from kr_a without knowing r_a.

E_2: In this transaction (i.e. step 2), P_b picks a random number r_b in the domain of $f()$, and computes:

$$kr_b = (k_b \times r_b^{-1}) \bmod n_t.$$

In addition, P_b forms its commitment co_b based on kr_a and r_b, which assures P_a that P_t can recover r_b from co_b. The formation and verification of co_b will be presented in Section 5. P_b then sends $E_{k_b}(D_b)$ and $E_{pk_a}(kr_b, co_b)$ to P_a.

After receiving P_b's message, P_a decrypts $E_{pk_a}(kr_b, co_b)$ with private key sk_a to get kr_b and co_b. P_a then confirms that $h(E_{k_b}(D_b))$ is the same as hd_b, and co_b is correct as will be detailed in Section 5. For any problem identified, P_a requires P_b to retransmit the message, or to restart the protocol. Otherwise, P_a executes transaction E_3.

E_3: In this transaction (i.e. step 3), P_a simply sends r_a to P_b. Upon arrival of r_a, P_b calculates:

$$k'_a = (kr_a \times r_a) \bmod n_t.$$

If $f(k'_a) = f(k_a)$ (see the assumption in Section 3), P_b decrypts $E_{k_a}(D_a)$ with k'_a to get D_a, and performs transaction E_4. Otherwise, P_b asks P_a to re-send the message.

E_4: In this transaction (i.e. step 4), P_b sends r_b to P_a which then calculates:

$$k'_b = (kr_b \times r_b) \bmod n_t.$$

If $f(k'_b) = f(k_b)$, P_a decrypts $E_{k_b}(D_b)$ with k'_b to obtain D_b. Otherwise, P_a requests P_b to retransmit the message.

When each party has received correct number r_a or r_b from the other party, the exchange is completed successfully, so no key recovery needs to be performed.

In case P_a cannot receive r_b from P_b through E_4 after having sent r_a to P_b through E_3, P_a can initiate the second sub-protocol, specified in Table 2 with the definitions of all the additional items used, to request P_t to perform key recovery. This sub-protocol is also illustrated as the UML sequence diagram in Fig. 2. The transactions of the sub-protocol are explained below:

Table 2
Off-line key recovery sub-protocol

Sub-protocol 2	
R_1. $P_a \rightarrow^A P_t$: $E_{pk_t}(sn, r_a, f(r_b), co_b, aca_{a,2})$ R_2. $P_t \rightarrow^A P_a$: sn, r_b	
Item	Definition
$aca_{a,2}$	P_a's anonymous contact address
$f(r_b)$	$= f(k_b) \times f(kr_b)^{-1} \bmod n_t$, computed by P_a

R_1: In this transaction, P_a transfers the items needed for the recovery of r_b to P_t. $f(r_b)$ in R_1 is calculated from $f(k_b)$ and kr_b by P_a. $aca_{a,2}$ is P_a's anonymous contact address to allow P_t to reply to the request, which could be different from $aca_{a,1}$ defined in Table 1 for better anonymity [7].

Having received P_a's request through transaction R_1, P_t decrypts $E_{pk_t}(sn, r_a, f(r_b), co_b, aca_{a,2})$ with private key sk_t to get items sn, r_a, $f(r_b)$, co_b and $aca_{a,2}$. P_t then computes r_b based on these items, and carries out necessary verifications, as will be detailed in Section 5. If the verifications are positive, P_t executes transaction R_2.

After executing R_2, P_t also publicises sn, r_a and r_b so that each of P_a and P_b can use sn to acquire r_a and r_b from P_t. This can prevent a party from repudiating receipt of r_a or r_b from P_t [16], which will be discussed further in Section 6. The practical implementation of such publication may vary, e.g. it can be a known query-reply service offered via a Web site.

R_2: In this transaction, P_t sends recovered r_b to P_a that can then compute P_b's key k_b as described earlier.

5. Key recovery

We now show how to produce P_b's commitment co_b. To achieve fairness and anonymity, the production of co_b needs to meet the following requirements:

(i) Recovery fairness: It is hard for P_a to compute r_b from co_b, and P_a cannot use co_b to obtain r_b from P_t without sending r_a to P_t. It is also hard for P_b to produce co_b such that P_a's verification shows that P_t can recover r_b from co_b, but the number r'_b actually recovered from co_b by P_t is different from r_b, i.e. $r_b \neq r'_b$.

(ii) Recovery anonymity: co_b does not include any information on the identities and locations of P_a and P_b.

The first requirement prevents not only P_a from unfairly gaining r_b from co_b, but also P_b from cheating P_a by stopping P_t recovering r_b. The second requirement ensures anonymity in the recovery process.

To produce co_b satisfying the above requirements, P_b first defines the following function:

$$p(x) = (r_b + r_b^{-1} \times x) \bmod n_t.$$

P_b then computes:

$$f(r_a) = f(k_a) \times f(kr_a)^{-1} \bmod n_t,$$
$$hv_b = h(sn, f(r_a), f(r_b)),$$
$$ep_1 = E_{pk_t}(p(hv_b)) = (r_b + r_b^{-1} \times hv_b)^{e_t} \bmod n_t,$$
$$ep_2 = E_{pk_t}(p(h(hv_b + 1))) = (r_b + r_b^{-1} \times h(hv_b + 1))^{e_t} \bmod n_t.$$

Here, sn, r_a, kr_a and r_b were defined in Section 4. P_b knows $f(k_a)$, P_t's public-key cryptosystem is based

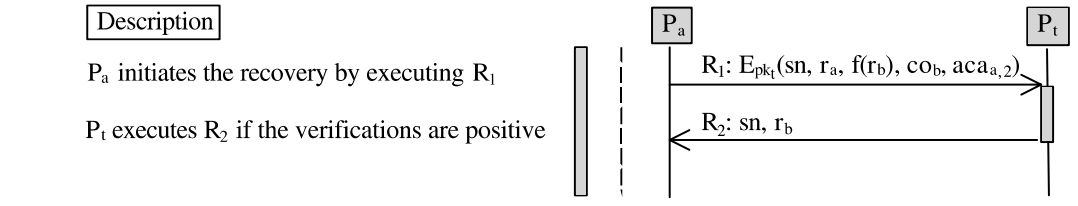

Fig. 2. Off-line key recovery sub-protocol.

on RSA, and P_t's public key is $pk_t = \{e_t, n_t\}$, as assumed in Section 3.

ep_1 and ep_2 actually link r_b to sn, $f(r_a)$ and $f(r_b)$ through hv_b. This prevents P_a from illegitimately altering these items, because any alteration will lead to a failure of the verifications to be performed by P_t.

P_b now defines its commitment as:

$$co_b = (ep_1, ep_2).$$

When receiving kr_b and co_b from P_b through transaction E_2, P_a computes $f(r_b) = f(k_b) \times f(kr_b)^{-1} \times \bmod n_t$ and $hv'_b = h(sn, f(r_a), f(r_b))$, and verifies:

$$E_{pk_t}(f(r_b) + 2 \times hv'_b + f(r_b)^{-1} \times hv'^2_b) = (r_b + r_b^{-1} \times hv'_b)^{2 \times e_t} \bmod n_t = f(ep_1),$$

$$E_{pk_t}(f(r_b) + 2 \times h(hv'_b + 1) + f(r_b)^{-1} \times h(hv'_b + 1)^2) = f(ep_2).$$

If this verification is positive, P_a is assured that P_t is able to compute r_b based on co_b together with sn, r_a and $f(r_b)$, as will be detailed below. It is thus secure for P_a to send r_a to P_b through transaction E_3.

In case a request for key recovery is received from P_a through transaction R_1 defined in Table 2, P_t decrypts $E_{pk_t}(sn, r_a, f(r_b), co_b, aca_{a,2})$ with its private key sk_t to obtain the five items, and then decrypts ep_1 and ep_2 in co_b with sk_t to recover dp_1 and dp_2 respectively. P_t can now compute $hv''_b = h(sn, f(r_a), f(r_b))$, and solve the following equations for r'_b and w:

$$\begin{cases} (r'_b + w \times hv''_b) \bmod n_t = dp_1, \\ (r'_b + w \times h(hv''_b + 1)) \bmod n_t = dp_2. \end{cases}$$

P_t then verifies that $f(r'_b) = f(r_b)$ and $(r'_b \times w) \times \bmod n_t = 1$. If the verification is positive, P_t is convinced that the request received from P_a is valid, as any illegitimate change to the items would result in a failure of the verification. P_t then sends r'_b to P_a which can recover P_b's key by computing $k_b = (kr_b \times r'_b) \bmod n_t$.

6. Protocol analysis

In this section we examine the fairness and anonymity of the protocol defined in Section 4 and the key recovery method presented in Section 5, respectively.

6.1. Fairness and anonymity of the document exchange protocol

We first demonstrate that the protocol can meet exchange fairness requirement (a) stated in Section 4, i.e. either each of P_a and P_b or neither of them can obtain the other's document. Suppose that P_b can obtain D_a, i.e. P_b has received r_a from either P_a or P_t. Note that it is hard for P_b to compute k_a from $kr_a (= (k_a \times r_a^{-1}) \bmod n_t)$ without knowing r_a. In this case, P_a has certainly received P_b's commitment co_b which enables P_a to obtain r_b (possibly with the help of P_t), i.e. P_a can obtain D_b. Here we assume that the key recovery method can satisfy recovery fairness and anonymity requirements (i) and (ii) defined in Section 5, which will be discussed in Section 6.2. Similarly, if P_a can obtain D_b, i.e. P_a has obtained r_b from either P_b or P_t, then P_b has received r_a from P_a, or can acquire r_a from P_t using session number sn, which allows P_b to obtain D_a.

Note that after obtaining co_b through transaction E_2 if P_a misbehaves by requesting P_t for key recovery without executing E_3, P_a cannot gain any advantage over P_b. This is because P_t accepts P_a's request only if P_a can provide correct r_a, and P_t also makes both r_a and r_b available for access by any party so that P_b can acquire r_a from P_t. This measure also offers non-repudiation of receipt. If a party falsely claims that it has not received r_a or r_b from P_t, then the party should repeat its request for r_a or r_b to P_t.

The above analysis demonstrates that the protocol can meet requirement (a).

We now show that the protocol can meet exchange anonymity requirement (b) stated in Section 4 as well. This is due to the fact that the contents of the transactions (i.e. E_1–E_4 and R_1–R_2) of the protocol do not include any information on the identities and locations of P_a and P_b, and communications between the parties involved can be conducted over anonymous channels. This means that if any one of P_a and P_b wishes to remain anonymous, it can conceal its location and identity from the other party and P_t. In other words, the

protocol can satisfy exchange anonymity requirement (b).

6.2. Fairness and anonymity of the key recovery method

We now demonstrate that the key recovery method can meet recovery fairness requirement (i) stated in Section 5. First, each of items ep_1 and ep_2 in co_b involves the encryption with P_t's public key, and it is hard for P_a to decrypt them without knowing P_t's private key. It is thus hard for P_a to compute r_b from co_b. If P_a sends co_b to P_t for the recovery of r_b without sending a correct r_a to P_t, the verifications conducted by P_t will fail. Consequently P_t will abort the recovery process, so P_a cannot obtain r_b from P_t. Secondly, the verification of co_b by P_a is based on computed result $f(r_b)$, i.e. $f(r_b) = f(k_b) \times f(kr_b)^{-1} \bmod n_t$, and it assures P_a that P_t can certainly derive r_b from ep_1 and ep_2 in co_b. If P_b produces co_b by using a number r'_b different from r_b in kr_b, then the verification of co_b by P_a will fail. This means that it is hard for P_b to successfully cheat P_a by producing an incorrect co_b. Therefore the method can satisfy recovery fairness requirement (i).

The method can also meet recovery anonymity requirement (ii) specified in Section 5 as the formation of co_b is based solely on sn, $f(r_a)$ and r_b which have no link to the identities and locations of P_a and P_b.

In addition, the key recovery method offers good simplicity and efficiency. Though there exist other recovery methods for fair exchange, e.g. those for signature recovery given in [5], they do not provide cost-effective anonymity and are mathematically complex. Our method is largely based on the well-known RSA algorithm, and its efficiency relies mainly on two encryptions for the generation and verification of commitment co_b, respectively, as well as two decryptions for recovery. This helps to improve the protocol's efficiency and applicability.

Moreover, as P_t only deals with random numbers r_a and r_b, the main role of P_t is to protect their integrity. This simplifies the security management and protection of P_t.

7. Conclusions

We have proposed a novel protocol for anonymous and fair document exchange between two parties P_a and P_b with the assistance of an off-line trusted third party P_t. At the heart of this protocol is the method for the generation and verification of P_b's commitment co_b assuring P_a that P_t can recover r_b from co_b which then allows P_a to compute P_b's key, in case P_a is unable to obtain r_b after having handed over r_a to P_b. This enables the protocol to achieve fairness effectively and efficiently. Also the key recovery conducted by P_t does not require any information about the identities, locations, exchanged documents and keys of P_a and P_b, so the impact of P_t's security on the protocol is weakened. This, coupled with anonymous communications between the parties involved, demonstrates the protocol's true anonymity. Of course, the protocol is also applicable to fair document exchange without anonymity protection by simply replacing anonymous communications with normal ones.

Acknowledgements

The work presented in this paper is part of the FIDES (Fair Integrated Data Exchange Services) project funded jointly by the UK Engineering and Physical Sciences Research Council (EPSRC) and Department of Trade and Industry (DTI). The project reference is LINK, GR/R55177/01.

We would like to thank the anonymous referees for their most constructive comments and suggestions.

References

[1] N. Asokan, M. Schunterand, M. Waidner, Optimistic protocols for fair exchange, Proc. ACM Conference on Computer and Communications Security, Switzerland, April 1997, pp. 6–17.

[2] N. Asokan, V. Shoup, M. Waidner, Asynchronous protocols for optimistic fair exchange, Proc. IEEE Symposium on Security and Privacy, Oakland, CA, May 1998, pp. 86–100.

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

[4] G. Ateniese, Efficient verifiable encryption (and fair exchange) of digital signatures, Proc. ACM Conference on Computer and Communications Security, Singapore, November 1999, pp. 138–146.
[5] F. Bao, R. Deng, W. Mao, Efficient and practical fair exchange protocols with off-line TTP, Proc. IEEE Symposium on Security and Privacy, Oakland, CA, May 1998, pp. 77–85.
[6] F. Bao, R. Deng, An efficient fair exchange protocol with an off-line semi-trusted third party, Proc. International Workshop on Cryptographic Techniques and E-Commerce, 1999, pp. 37–47.
[7] D.L. Chaum, Untraceable electronic mail return addresses and digital pseudonyms, Communications of the ACM 24 (1981) 84–88.
[8] L. Chen, Efficient fair exchange with verifiable confirmation of signatures, in: Proc. Advances in Cryptology—ASIACRYPT '98, Springer, Berlin, 1998, pp. 286–299.
[9] B.P. Douglass, Real-time—UML developing efficient objects for embedded systems, Addison Wesley Longman, Reading, MA, 1998.
[10] S. Even, O. Goldreich, A. Lempel, A randomised protocol for signing contracts, Communications of the ACM 28 (1985) 637–647.
[11] M. Franklin, M. Reiter, Fair exchange with a semi-trusted third party, Proc. ACM Conference on Computer and Communications Security, Zurich, Switzerland, April 1997, pp. 1–5.
[12] T. Okamoto, K. Ohta, How to simultaneously exchange secrets by general assumptions, Proc. ACM Conference on Computer and Communication Security, New York, USA, 1994, pp. 184–192.
[13] I. Ray, I. Ray, An optimistic fair exchange e-commerce protocol with automated dispute resolution, in: Proc. First International Conference on Electronic Commerce and Web Technologies, EC-Web 2000, Lecture Notes in Computer Science, vol. 1875, Springer, Berlin, 2000, pp. 84–93.
[14] I. Ray, I. Ray, An anonymous fair exchange e-commerce protocol, Proc. First International Workshop on Internet Computing and E-Commerce, San Francisco, CA, April 2001, pp. 1790–1797.
[15] R.L. Rivest, A. Shamir, L.M. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM (1978) 120–126.
[16] N. Zhang, Q. Shi, Achieving non-repudiation of receipt, The Computer Journal 39 (1996) 844–853.

Ning Zhang is a Lecturer at the Department of Computer Science, University of Manchester, UK. She received her Ph.D. in Electronic Engineering from the University of Kent at Canterbury, UK. Her research interests include computer networks, mobile computing, and information and e-commerce security. She is supervising a number of research projects on these subjects, which are funded by various funding sources, including the UK Engineering and Physical Sciences Research Council (EPSRC) and Department of Trade and Industry (DTI).

Qi Shi received his Ph.D. in Computing from Dalian University of Technology, P.R.C. He worked as a research associate for the Department of Computer Science at University of York in the UK. Dr. Shi then joined the School of Computing & Mathematical Sciences at Liverpool John Moores University in the UK, and he is currently a Reader. His current research interests include computer security and formal methods.

Madjid Merabti is Deputy Director and Head of Research, School of Computing & Mathematical Sciences, Liverpool John Moores University, UK. He is a graduate of Lancaster University in the UK. He has over 10 years experience in conducting research and teaching in the areas of Distributed Multimedia Systems (Computer Networks, Mobile Computing, and Computer Security). Prof. Merabti is widely published in these areas and leads the Distributed Multimedia Systems Research Group which has a number of Government and industry supported research projects. Current projects include Multimedia Networking, Mobile Networks Security and Privacy Architectures and Protocols, Networked Appliances and Mobile Computing Environments. He is the programme chair for the 5th IEEE International Workshop on Networked Appliances, October 2002.