A Security Protocol for Certified E-Goods Delivery

Aleksandra Nenadić, Ning Zhang, Stephen Barton
Department of Computer Science, University of Manchester, UK
{anenadic, nzhang, s.k.barton}@cs.man.ac.uk

Abstract

This paper presents an efficient security protocol for certified e-goods delivery with the following features: (1) ensures strong fairness, (2) ensures non-repudiation of origin and non-repudiation of receipt, (3) allows the receiver of an e-goods to verify, during the protocol execution, that the e-goods he is about to receive is the one he is signing the receipt for, (4) does not require the active involvement of a fully trusted third party, but rather an off-line and transparent semi-trusted third party (STTP) only in cases of unfair behaviour by any party, and (5) provides confidentiality protection for the exchanged items from the STTP.

1. Introduction

One of the principal activities in e-commerce is concerned with fair data exchange between business parties, which is required when the exchanged data contain valuable information. Certified e-goods delivery is such an example activity, where valuable e-goods are exchanged for an acknowledgement of its reception. The exchange in this case must ensure two properties: firstly, *fairness*, i.e. a recipient will receive the e-goods together with a proof of its origin from the sender if and only if the sender receives the acknowledgement of reception from the recipient, and, secondly, *content/quality assurance*, i.e. the received e-goods indeed match the expected description/quality.

Over the past few years, researchers have been working on the design of fair non-repudiation protocols to achieve certified delivery [1, 4, 7, 8, 10, 13, 14]. This work is mainly focused on certified e-mail delivery, while certified e-goods delivery has not been addressed properly. The main difference between an e-mail and an e-goods in the context of certified delivery lies in content/quality assurance. With certified e-mail delivery, the emphasis is on the generation of a piece of evidence that can prove the reception of a specific e-mail by the recipient; whether or not the content of the e-mail matches with the recipient's expectations does not matter much. For certified e-goods delivery, on the other hand, a signed receipt should guarantee not only the reception of an e-goods, but also its content/quality, as a mismatch between the expected and actual contents often has financial implications and should be detected before the exchange process is over.

This paper is focused on the design of a fair non-repudiation RSA-CEGD protocol to support certified e-goods delivery. The protocol can be applied in numerous e-commerce applications where e-goods represent software, video or audio content, images, electronic payments, etc. An idea for achieving certified e-goods delivery efficiently and securely is to firstly encrypt the e-goods with a symmetric key and transfer the encrypted e-goods to the recipient prior to the exchange, and then to engage in a fair exchange of the decryption key for the recipient's signature (i.e. receipt). In order to ensure the *strong fairness* [1] and *content/quality assurance* properties for the exchange, the RSA-CEGD protocol utilises two main concepts - *verifiable and recoverable encrypted signature* (VRES) and *joint e-goods and key certification*.

The VRES represents a signature encrypted in such a way that a receiver of the VRES is assured that it indeed contains the correct signature without obtaining any information about the signature itself (*verifiability*). He is also assured that a designated trusted third party can help to recover the original signature from the VRES, in case the original signature sender refuses to release his receipt after receiving the e-goods (*recoverability*). In detail, the VRES is applied in the following way to guarantee strong fairness in certified e-goods delivery: the recipient of e-goods generates the VRES of his signature (i.e. receipt) and then uses it to secure the release of the sender's key (for the decryption of the e-goods). Upon successful verification of the VRES, the sender of the e-goods is assured that it is secure for him to release his key first. The recipient can then respond by sending his original signature, so as to complete the exchange successfully without any involvement of any third party. Should the recipient refuse to surrender his signature after receiving the e-goods' decryption key, the sender can resort to the agreed third party, which recovers the original signature from the VRES and restores fairness.

The concept of *joint e-goods and key certification* is utilised in the protocol to enable the recipient to verify the correctness of the encrypted e-goods and the decryption key during a protocol execution. In this way, content/quality assurance of the e-goods can be achieved - the recipient is assured that the e-goods obtained by decryption with the received decryption key at the end of the exchange will indeed match with what is expected. The linkage between the original e-goods, encrypted e-goods and the decryption key should be validated and certified by a certification authority prior to the exchange.

The major novel contributions of this paper can be summarized as follows. Although the idea of digital content validation/certification is not entirely new on the Internet (e.g. Microsoft's Authenticode), and has been utilised in fair e-purchase [9] and fair document exchange protocols [11], this paper is the first, to the best of authors' knowledge, that has presented a protocol incorporating the concept of joint e-goods and key certification to allow certified e-goods delivery. The protocol utilises a novel and efficient RSA-based VRES method, employs an off-line STTP, which is invoked only in cases when the two exchanging parties cannot reach a fair completion of the exchange themselves, the receipts are standard RSA signatures. The design of the STTP is such that security requirements imposed on it are much simplified in comparison with a fully trusted third party.

The rest of the paper is organised as follows. Section 2 states the requirements that the RSA-CEGD protocol aims to satisfy, together with the notation and assumptions used in the protocol design. Section 3 gives an overview of the protocol. The main cryptographic primitive VRES is described in Section 4. The protocol is formally presented in Section 5. Section 6 provides the protocol analysis, and Section 7 presents a comparison of the protocol with related work. Finally, Section 8 outlines our conclusions.

2. Preliminaries

2.1 Requirements

The RSA-CEGD protocol is designed to satisfy the following requirements.

(S1) *Non-repudiation of origin* - the recipient is provided with a proof that the sender is indeed the originator of the e-goods.

(S2) *Non-repudiation of receipt* - the sender is provided with a proof that the intended recipient has indeed received the e-goods.

(S3) *Strong fairness* - iff the sender has obtained the receiver's receipt or can obtain it with the assistance of a STTP, then the receiver has obtained the sender's e-goods or can obtain it with the assistance of the STTP.

(S4) *E-goods content/quality assurance* – the receiver of the e-goods can verify, during the protocol execution, that the e-goods he is about to receive is the one he is signing the receipt for.

(S5) *E-goods and receipt confidentiality* - e-goods to be delivered and the corresponding receipt are not disclosed to any external party, including the STTP.

(S6) *Transparency of the STTP* - the participation of the STTP in the protocol is transparent, i.e. the signature recovered by the STTP is indistinguishable from that sent by the original signer.

2.2 Notation

The following notation is used to describe the RSA-CEGD protocol.

- $E_k(x)$ expresses a ciphertext of a data item x encrypted with a key k. $E_k(x)$ is computed using a symmetric cryptosystem if the corresponding decryption key is the same as k, or an RSA public-key cryptosystem otherwise.
- $h(x)$ is a strong-collision-resistant one-way hash function.
- x, y denotes the concatenation of data items x and y.

2.3 Assumptions

The following assumptions are used in the design of the RSA-CEGD protocol.

- Party P_a has valuable e-goods, denoted as D_a, and a symmetric key k_a for encryption and decryption of D_a. P_a wishes to send D_a to party P_b in exchange for party P_b's receipt for D_a.
- P_a and P_b have agreed to employ an off-line STTP P_t to help them with the exchange if they cannot reach a fair completion of the exchange themselves.
- The content of D_a is certified by an authority, which has issued a certificate $CertD_a = (desc_a, hd_a, h_a, ek_a, sign_{at})$ to P_a for D_a. Here, $desc_a$ is the content summary of D_a, $hd_a = h(E_{k_a}(D_a))$ is the hash value of the encryption of D_a with key k_a, $h_a = h(D_a)$ is the hash value of D_a, $ek_a = E_{pk_a}(k_a)$ is the encryption of k_a with P_a's public key pk_a, and $sign_{at}$ is the authority's signature on items $desc_a$, hd_a, h_a and ek_a. Certificate links the encrypted D_a, its description $desc_a$ and the decryption key k_a, and the signature $sign_{at}$ represents the authority's approval that if an encrypted data $E_{k_a}(D_a)$ and the key k_a meet the

conditions $h(E_{k_a}(D_a)) = hd_a$ and $E_{pk_a}(k_a) = ek_a$, then decryption of $E_{k_a}(D_a)$ using the key k_a will recover D_a with its contents matching the description in $desc_a$. This certification process is explained in more details in [11].

- Party P_b has obtained the encrypted e-goods $E_{k_a}(D_a)$ and $CertD_a$ from P_a prior to the exchange.
- Every party P_i ($i \in \{a, b, t\}$) has a pair of public and private RSA keys, expressed as $pk_i = (e_i, n_i)$ and $sk_i = (d_i, n_i)$, where public key pk_i ($i \in \{a, b, t\}$) is certified by a certification authority and is known by all the other parties.
- Party P_b's receipt for D_a, denoted as $receipt_b$, is represented by P_b's RSA signature on D_a, i.e.
  $receipt_b = (h_a)^{d_b} \bmod n_b = E_{sk_b}(h_a)$.
- P_b has obtained an additional RSA public-key certificate $C_{bt} = (pk_{bt}, w_{bt}, s_{bt})$ for an additional public-key pk_{bt} issued by P_t prior to the exchange. The purpose of this certificate is to establish a secret private key sk_{bt} shared between P_b and P_t, so that P_b can use this secret key to create the VRES, and P_t can use it recover the VRES in case of dispute. $pk_{bt} = (e_{bt}, n_{bt})$ in certificate C_{bt} is the RSA public key related to the secret key $sk_{bt} = (d_{bt}, n_{bt})$, where n_{bt} is a product of two distinct large primes chosen by P_t, and e_{bt} is required to be the same as e_b, i.e. $e_b = e_{bt}$. w_{bt} in C_{bt} is defined as $w_{bt} = (h(sk_t, pk_{bt})^{-1} \times d_{bt}) \bmod n_{bt}$, where sk_t is P_t's private key. s_{bt} in certificate C_{bt} is P_t's RSA signature on $h(pk_{bt}, w_{bt})$, i.e. $s_{bt} = E_{sk_t}(h(pk_{bt}, w_{bt}))$. In this way, P_t has no need to store sk_{bt}, as it can compute it from w_{bt}, i.e. $d_{bt} = (h(sk_t, pk_{bt}) \times w_{bt}) \bmod n_{bt}$. P_t only needs to issue one certificate C_{bt} for P_b (for instance when party P_b registers with P_t).
- The VRES recovery method is based on the following theory (for the proof see [9]):

Theory of cross-decryption: Let n_1 and n_2 be relatively prime and bases of two RSA cryptosystems, and $e_1 = e_2 = e$ the corresponding public-key exponents. For any two messages m and m', such that $m < \min(n_1, n_2)$ and $m' < \min(n_1, n_2)$, the following holds:

$(m^e \bmod (n_1 \times n_2)) \bmod n_1 = (m')^e \bmod n_1$ if and only if $m = m'$, and

$(m^e \bmod (n_1 \times n_2)) \bmod n_2 = (m')^e \bmod n_2$ if and only if $m = m'$.

- Party P_a initiates the RSA-CEGD protocol. Communication channels to/from P_t are *resilient*, i.e. all messages inserted in such channels will eventually reach the intended recipients. No reliability assumptions are made about the channel between P_a and P_b, i.e. it may be *unreliable* and messages may be lost. Channels between all the parties are authenticated and confidential.

3. The RSA-CEGD Protocol Overview

Parties P_a and P_b first attempt to perform the exchange themselves. To secure the release of party P_a's key k_a for the decryption of D_a, P_b produces and transfers to P_a a verifiable and recoverable encryption (VRES) of his receipt $receipt_b$ for P_a's D_a, along with his authorisation for the receipt recovery s_b. Authorisation s_b is used by P_a to request P_t to recover P_b's $receipt_b$ from the VRES in case of dispute, and states the conditions under which P_t should recover $receipt_b$ for P_a. P_a verifies the correctness of P_b's VRES and authorisation s_b, and, if satisfactory, P_a is convinced that P_t can help to recover the encrypted receipt, should it be necessary, and that he has the correct authorisation s_b for the receipt recovery, so it is secure for P_a to release the key k_a to P_b first. P_b can then respond by letting P_a have $receipt_b$, thus completing the exchange process successfully and without any involvement of P_t. In case where P_a fails to obtain $receipt_b$ after handing the key k_a to P_b, P_a can contact P_t for the receipt recovery. After P_t has recovered P_b's receipt from the VRES, a fair completion of the exchange process is achieved.

4. Verifiable and Recoverable Encrypted Signature (VRES)

In this Section we describe the VRES, the cryptographic primitive that is applied to encrypt P_b's $receipt_b$, and its generation, verification and recovery.

4.1 VRES Generation

To produce the VRES of $receipt_b$ for P_a's D_a, denoted as (y_b, x_b, xx_b), party P_b chooses a random number r_b that is relatively prime to each of n_b and n_{bt}, and computes:

$y_b = r_b^{e_b} \bmod (n_b \times n_{bt})$,

$x_b = (r_b \times E_{sk_b}(h_a)) \bmod n_b = (r_b \times (h_a)^{d_b}) \bmod n_b =$

$= (r_b \times receipt_b) \bmod n_b$,

$xx_b = (r_b \times E_{sk_{bt}}(h(y_b))) \bmod n_{bt} = (r_b \times (h(y_b))^{d_{bt}}) \bmod n_{bt}$.

Here, e_b and d_{bt} are exponents of public key pk_b and private key sk_{bt}, respectively. h_a (= $h(D_a)$) represents a hash value of D_a, and is included in D_a's certificate.

Note that $y_b \bmod n_b = (r_b^{e_b} \bmod (n_b \times n_{bt})) \bmod n_b = r_b^{e_b} \bmod n_b = E_{pk_b}(r_b)$, and similarly $y_b \bmod n_{bt} = E_{pk_{bt}}(r_b)$, so y_b can be decrypted using either P_b's

private key sk_b or secret key sk_{bt}, shared between P_b and P_t, to recover r_b, according to the theory of cross-decryption (see Section 2.3). In the process of the VRES generation, P_b's $receipt_b$ is effectively encrypted in x_b using random r_b, and r_b is encrypted in y_b, and r_b can only be recovered by P_b or P_t.

4.2 VRES Verification

The verification of P_b's VRES is defined as follows.

Verification 1:

(*a*) *Check the correctness of signature s_{bt} contained in P_b's certificate $C_{bt} = (pk_{bt}, w_{bt}, s_{bt})$, i.e. decrypt s_{bt} with P_t's public key pk_t to gain a hash value hv', and confirm that $h(pk_{bt}, w_{bt}) = hv'$.*

(*b*) *Confirm that* $x_b^{e_b} \bmod n_b = (r_b \times h_a^{d_b})^{e_b} \bmod n_b = (y_b \times h_a) \bmod n_b$.

(*c*) *Confirm that* $xx_b^{e_b} \bmod n_{bt} = (r_b \times E_{sk_{bt}}(h(y_b)))^{e_b} \bmod n_{bt} = (r_b \times (h(y_b))^{d_{bt}})^{e_b} \bmod n_{bt} = (y_b \times h(y_b)) \bmod n_{bt}$.

Here, $e_b = e_{bt}$. The purpose of Verification 1 is to convince P_a that P_b's VRES contains the correct $receipt_b$ without disclosing it to P_a, and that P_t can decrypt the VRES (i.e. recover random number r_b) to recover $receipt_b$. In detail, Verification 1(*a*) makes sure that certificate C_{bt} is valid so that P_t can recover the secret key sk_{bt} related to the public key pk_{bt} in C_{bt}. Verification 1(*b*) confirms that item x_b contains P_b's correct $receipt_b$. Verification 1(*c*) together with (*b*) ensures that the same number r_b is used in the computations of y_b, x_b and xx_b, and that the modulus operation in y_b is based on $n_b \times n_{bt}$, so that P_t can decrypt y_b with the secret key sk_{bt} to obtain r_b.

4.3 VRES Recovery

To recover P_b's encrypted receipt, P_t first has to derive the shared private key $sk_{bt} = (d_{bt}, n_{bt})$ from P_b's certificate $C_{bt} = (pk_{bt}, w_{bt}, s_{bt})$ using its private key sk_t, i.e. P_t computes:

$$d_{bt} = (h(sk_t, pk_{bt}) \times w_{bt}) \bmod n_{bt}.$$

P_t then uses the derived secret key sk_{bt} to decrypt y_b $mod\ n_{bt} = E_{pk_{bt}}(r_b)$ to recover r_b. Number r_b can now be used by party P_a to recover $receipt_b$ from x_b as follows:

$$receipt_b = (r_b^{-1} \times x_b) \bmod n_b.$$

5. The RSA-CEGD Protocol

The RSA-CEGD is a protocol suite consisted of two protocols – the exchange protocol and the recovery protocol. The exchange protocol handles the case of a normal exchange between parties P_a and P_b without P_t's involvement. The recovery protocol deals with cases when the exchange protocol has failed to complete successfully due to a party's misbehavior or a network failure, during which P_t is invoked to recover the receipt and restore fairness.

5.1 The Exchange Protocol

It is assumed that P_b has received P_a's encrypted D_a (i.e. $E_{k_a}(D_a)$) and D_a's certificate ($desc_a$, hd_a, h_a, ek_a, $sign_{at}$) prior to the exchange. The exchange protocol is shown in Table 1, and the definitions of the protocol's items in Table 2.

Table 1. The exchange protocol

(E1): $P_a \rightarrow P_b$: x_a, $E_{sk_a}(h_a)$
(E2): $P_b \rightarrow P_a$: x_b, xx_b, y_b, s_b, C_{bt}
(E3): $P_a \rightarrow P_b$: r_a
(E4): $P_b \rightarrow P_a$: r_b

Table 2. Definitions of the items

$x_a = (r_a \times k_a) \bmod n_a$: encryption of P_a's key k_a with random number r_a;
$h_a = h(D_a)$: hash value of e-goods D_a to be delivered;
$E_{sk_a}(h_a)$: P_a's signature on D_a;
r_a: P_a's random prime for the encryption of key k_a;
$y_a = E_{pk_a}(r_a)$: RSA encryption of number r_a with key pk_a;
r_b: P_b's random prime for the generation of (y_b, x_b, xx_b);
(y_b, x_b, xx_b): P_b's VRES, where $y_b = r_b^{e_b} \bmod (n_b \times n_{bt})$: encryption of r_b with P_b's public key pk_b, also recoverable by P_t, $x_b = (r_b \times (h_a)^{d_b}) \bmod n_b = (r_b \times receipt_b) \bmod n_b$: encryption of $receipt_b$ with r_b, $xx_b = (r_b \times E_{sk_{bt}}(h(y_b))) \bmod n_{bt}$;
C_{bt}: P_b's RSA public-key certificate issued by P_t, $pk_{bt} = (e_{bt}, n_{bt})$, $sk_{bt} = (d_{bt}, n_{bt})$: public and private RSA keys related to C_{bt} with $e_{bt} = e_b$;
$s_b = E_{sk_b}(h(C_{bt}, y_b, y_a, P_a))$: P_b's authorisation token;

Step (E1): For the agreed exchange, party P_a chooses a random prime number $r_a < n_a$, and uses r_a to encrypt D_a's decryption key k_a as follows:

$$x_a = (r_a \times k_a) \bmod n_a.$$

P_a then transmits x_a together with its digital signature $E_{sk_a}(h_a)$ on D_a to P_b. The latter will serve as a non-repudiable proof of origin of D_a.

Step (E2): Upon receipt of the two items, P_b verifies P_a's signature as defined in Verification 2.

Verification 2: Decrypt the signature $E_{sk_a}(h_a)$ with P_a's public key pk_a to obtain a hash value h_a', and compare it with the one contained in D_a's certificate (i.e. h_a).

If the two hash values are not equal, P_b may ask P_a to re-send message (E1) or terminate the protocol execution. Otherwise, P_b computes y_a as follows:

$$y_a = (E_{pk_a}(x_a) \times ek_a^{-1}) \bmod n_a.$$

Here, pk_a is P_a's public key and $ek_a = E_{pk_a}(k_a)$ is from D_a's certificate. As $y_a = (E_{pk_a}(x_a) \times ek_a^{-1}) \bmod n_a = ((r_a \times k_a)^{e_a} \times k_a^{-e_a}) \bmod n_a = r_a^{e_a} \bmod n_a = E_{pk_a}(r_a)$, y_a actually represents the RSA encryption of number r_a using P_a's public key pk_a. P_b then produces and transfers to P_a the two items – the VRES (y_b, x_b, xx_b), described in Section 4.1, and authorisation s_b, which is simply P_b's RSA signature on items C_{bt}, y_b, y_a, P_a, i.e.:

$$s_b = (h(C_{bt}, y_b, y_a, P_a))^{d_a} \bmod n_a.$$

C_{bt} in authorisation s_b is P_b's certificate, y_b represents P_b's encrypted r_b, y_a specifies what P_b expects from P_a with respect to key k_a, and P_a is P_a's identity. s_b represents P_b's conditional authorisation stating that *P_t can recover r_b from y_b for P_a (which will enable P_a to derive receipt$_b$ from x_b) if and only if P_a provides P_t with an item r_a, such that $E_{pk_a}(r_a) = y_a$ (which will allow P_b to compute P_a's key k_a from x_a).*

Step (E3): P_a performs Verification 1 to check the correctness of P_b's VRES (y_b, x_b, xx_b), and Verification 3 below to check P_b's authorisation s_b.

Verification 3: Decrypt s_b with P_b's key pk_b to recover a hash value hv'', and confirm $h(C_{bt}, y_b, y_a, P_a) = hv''$.

If Verifications 1 and 3 are both positive, it is secure for P_a to send r_a to P_b. Otherwise, P_a can terminate the protocol execution without violating fairness, or ask P_b to re-send message (E3).

Step (E4): Upon receipt of r_a from P_a, P_b uses it to derive P_a's key k_a from x_a received earlier:

$$k_a = (r_a^{-1} \times x_a) \bmod n_a.$$

P_b then examines the correctness of key k_a using Verification 4.

Verification 4: Confirm that $E_{pk_a}(k_a) = ek_a$, where ek_a is contained in D_a's certificate.

If Verification 4 is positive, P_b uses k_a to decrypt $E_{k_a}(D_a)$ to obtain D_a, and sends number r_b to P_a. Otherwise, P_b can terminate the protocol execution without violating fairness, or ask P_a to re-send message (E4). Upon receipt of r_b, P_a uses it to derive P_b's *receipt$_b$* from x_b received earlier:

$$receipt_b = (r_b^{-1} \times x_b) \bmod n_b.$$

P_a then goes through Verification 5 to verify the correctness of *receipt$_b$*.

Verification 5: Confirm that $E_{pk_b}(receipt_b) = h_a$, i.e. decrypt receipt$_b$ with P_b's public key pk_b to gain a hash value h_a'', and confirm that $h_a = h_a''$, where $h_a = h(D_a)$ is the hash value of D_a.

If Verification 5 is positive, the exchange protocol is completed successfully. That is, party P_a has obtained P_b's *receipt$_b$* and party P_b has obtained P_a's D_a. If Verification 5 is negative, or P_a fails to receive r_b from P_b, P_a can request P_t for the receipt recovery.

5.2 The Recovery Protocol

The recovery protocol (shown in Table 3) can be invoked by party P_a only.

Table 3. – The recovery protocol

(R1): $P_a \rightarrow P_t$: C_{bt}, y_b, s_b, y_a, r_a
(R2): $P_t \rightarrow P_a$: r_b
(R3): $P_t \rightarrow P_b$: r_a

Step (R1): P_a transfers the items C_{bt}, y_b, s_b, y_a and r_a to P_t to request the receipt recovery. P_t performs the following verification.

Verification 6:

(*a*) *Check the correctness of P_b's authorisation s_b using Verification 3 defined above.*

(*b*) *Confirm that $E_{pk_a}(r_a) = y_a$.*

Verification 6(*b*) ensures the correctness of the received r_a. If Verification 6 is negative, P_t rejects P_a's request. Otherwise, P_t recovers r_b as described in Section 4.3.

Step (R2): P_t then sends r_b to P_a, who uses it to recover *receipt$_b$* from x_b received earlier, i.e.:

$$receipt_b = (r_b^{-1} \times x_b) \bmod n_b.$$

Step (R3): P_t also forwards r_a to P_b, who may use it to recover P_a's key k_a from x_b received earlier, i.e.:

$$k_a = (r_a^{-1} \times x_a) \bmod n_a.$$

6. The RSA-CEGD Protocol Analysis

In this Section, we analyse the security of the protocol. We first analyse the security of the VRES (y_b, x_b, xx_b) and the "encryption" x_a of P_a's key k_a.

$y_b = r_b^{e_b} \bmod (n_b \times n_{bt})$ is a minor variation of RSA encryption, so it is hard for any party P_o ($\notin \{P_b, P_t\}$) to decrypt y_b to obtain r_b. It is also hard for P_o to factor $xx_b = (r_b \times E_{sk_{bt}}(h(y_b))) \bmod n_{bt}$ to obtain r_b from xx_b. Similarly, it is hard for P_o ($\neq P_b$) to factor $x_b = (r_b \times receipt_b) \bmod n_b$ to gain *receipt$_b$*. Therefore, P_o cannot illegitimately obtain r_b nor *receipt$_b$* from (y_b, x_b, xx_b). It is also hard for P_b to forge the VRES, i.e. use different $receipt_b' = (h_a')^{d_b} \bmod n_b$ in the VRES generation, as P_a can detect this deceitfulness through Verification 1.

Similar discussion can be applied to encryption x_a of P_a's key k_a, i.e. it is hard for any party P_o ($\neq P_a$) to factor $x_a = (r_a \times k_a) \bmod n_a$ to obtain key k_a.

The following discussion shows that the protocol meets the requirements set in Section 2.1.

- *Non-repudiation and fairness*: Suppose that P_b has obtained P_a's D_a, i.e. P_b has received the key k_a for the decryption of D_a in step (E3) or in step (R3). In this case, P_a has certainly got the correct items from P_b in step (E2). Consequently, P_a can obtain r_b from P_b in step (E4), or from P_t in step (R2). After obtaining r_b, P_a can use it to derive P_b's receipt from x_b. Similarly, suppose that P_a has obtained $receipt_b$, i.e. P_a has received the correct items from P_b in step (E2), and r_b in step (E4) or from P_t in step (R2). This implies that P_b has received the correct r_a for decryption of the key k_a from P_a in step (E3) or from P_t in step (R3). Therefore, at the end of the protocol, either party P_b will receive D_a and party P_a will receive $receipt_b$, or neither of them will receive anything useful. Party P_a must also provide the correct signature on D_a in step (E1), as a proof of origin of D_a. Therefore the protocol meets the requirements (S1)-(S3).
- *E-goods content/quality assurance*: Based on the certificate $CertD_a$ issued by a Certification Authority party P_b can verify the correctness of the decryption key k_a during the protocol execution, and he trusts this Authority to perform the certification correctly. In this way, P_b is assured that the e-goods D_a he will obtain at the end of the protocol (by decryption with the key k_a) will indeed match with the description from the $CertD_a$ (S4).
- *E-goods and receipt confidentiality*: During the recovery process P_t deals only with random numbers r_a and r_b, while e-goods D_a, key k_a, $receipt_b$, and numbers x_a and x_b (which can be used to derive k_a and $receipt_b$) are not disclosed to P_t. Thus, the privacy of the exchanged e-goods and the corresponding receipt is preserved (S5).
- *Transparency of the STTP*: It is evident that the structure of the receipt received from P_b in the exchange protocol is not different from that recovered by P_t during the recovery protocol, i.e. the received receipt does not reveal whether or not P_t has been involved in the exchange process (S6).

7. Comparison with Related Work

In this section, we summarise the main characteristics of the RSA-CEGD protocol and compare it with related work. To the best of authors' knowledge, this paper presents the first protocol for certified e-goods delivery with an embedded e-goods content/quality assurance. As certified e-mail protocols with off-line third party are the most related to ours, we focus our comparison to this class of protocols.

In certified e-mail protocols [1, 14], a proof of receipt is represented by a token consisted of several items and is not a standard signature. Also, in cases when a third party has to be invoked, it generates a proof of receipt in the name of the original signer, which has the same legal value but is structurally different from the one produced by the original signer. This means that the third party in these protocols is not transparent. In contrast to this, receipts received in the RSA-CEGD protocol are standard RSA-based signatures, and the STTP's participation is transparent.

The RSA-CEGD protocol is designed in such a way that only the sender is actively involved in the receipt recovery, while the recipient only takes a passive role in this process. This reduces the communication load on the recipient and safeguards him from potential denial-of-service attacks from malicious senders. Protocols [1, 8, 14] require both the sender and the recipient to actively participate in dispute resolution, and, in order to prematurely terminate the normal exchange protocol they have to contact the third party and execute an abort protocol. In other words, these protocol suites comprise 3 or 4 protocols each, which increases the communication overheads.

The VRES principle, on which the RSA-CEDG protocol is based, has been so far mainly utilised in fair signature exchange protocols [2, 3, 5, 6, 12]. It has been recently applied to two certified delivery protocols [4, 8].

Markowitch and Saeednia presented a certified e-goods delivery protocol in [8] with a non-interactive VRES scheme and receipts based on GPS signature scheme, which is not a signature scheme often used in practise. During the recovery process, the third party has to verify that the e-goods' contents match the description, but this process has not been fully clarified and various problems may arise when performing this verification automatically, as it is sometimes difficult to precisely describe the e-goods or its features.

Ateniese and Nita-Rotaru's protocol [4] is a certified e-mail delivery protocol, so no quality/content assurance is required, whereas our protocol is designed for certified e-goods delivery and therefore needs to meet this additional security requirement. Protocol [4] employs an interactive VRES scheme, which is less efficient, but the receipts are RSA-based. Both our protocol and Ateniese and Nita-Rotaru's protocol have an initialisation stage for party P_b and the TTP to agree on a shared secret that is used in the recovery process, and during which P_b receives a certificate from the TTP for this secret. In [4], the TTP is required to store

and safe-keep the shared secret, whereas in our protocol the TTP does not need to store anything – the secret can be computed from P_b's certificate C_{bt}, so the security and storage requirements placed on the TTP are reduced.

Table 4. Comparison of the protocols' efficiency

	Our	[4]	[8]
# exp. in VRES generation	3	3	2
# exp. in VRES verification	3	4	2
# exp. in VRES recovery	1	5	1
# exp. in the exch. protocol	11	14	16
# exp. in the rec. protocol	3	7	4/4*
# mess. in the exch. protocol	4	7	4
# mess. in the rec. protocol	3	3	3/3*

*Numbers shown are for P_a's/ P_b's recovery protocols

Both protocols [4, 8] employ a transparent third party, but neither satisfies the confidentiality requirement (S4) for the e-goods and receipt. The third party in [4, 8] also must be fully trusted, whereas in our protocol the third party is semi-trusted.

The efficiency of our protocol and protocols [4, 8] is evaluated in terms of the number of protocol messages and the number of expensive computations involved in the formation and verification of the messages. Expensive computations here refer to modular exponentiations. The evaluation is presented in Table 4 and demonstrates that our protocol imposes less communicational and computational costs on the participants of the protocol.

8. Conclusions

This paper has presented a fair non-repudiation e-commerce protocol for certified e-goods delivery. The protocol is based on two important methods: the VRES method, which enables the protocol to achieve strong fairness, and the e-goods and key certification method, which prevents a dishonest party from using some junk data in exchange for the receipt. The e-goods and the receipt exchanged enjoy the confidentiality protection, and the protocol places only weak security requirements on the STTP. Our future work will include the formal verification and prototyping of the RSA-CEGD protocol.

References

[1] Asokan, N., Shoup, V., Waidner, M., Asynchronous Protocols for Optimistic Fair Exchange, Proceedings of the IEEE Symposium on Security and Privacy, 1998, pp. 86-100.

[2] Asokan, N., Schunter, M., Waidner, M., Optimistic Fair Exchange of Digital Signatures, IEEE Journal on Selected Areas in Communications 18, 2000, pp. 593-610.

[3] Ateniese, G., Efficient Verifiable Encryption (and Fair Exchange) of Digital Signatures, Proceedings of ACM Conference on Computer and Communications Security, 1999, pp. 138-146.

[4] Ateniese, G., Nita-Rotaru, C., Stateless-recipient Certified E-mail System Based on Verifiable Encryption, Proceedings of the Topics in Cryptology, The Cryptographers' Track at the RSA Conference 2002, LNCS, vol. 2271, Springer-Verlag, Berlin, Germany, 2002, pp. 182-199.

[5] Bao, F., Deng, R., Mao, W., Efficient and Practical Fair Exchange Protocols with Off-line TTP, Proceedings of IEEE Symposium on Security and Privacy, 1998, pp. 77-85.

[6] Chen, L., Efficient Fair Exchange with Verifiable Confirmation of Signatures, Proceedings of Advances in Cryptology - ASIACRYPT '98, LNCS, vol. 1514, Springer-Verlag, Berlin, Germany, 1998, pp. 286-299.

[7] Deng, R. H., Gong, L., Lazar, A.A., Wang, W., Practical Protocols for Certified Electronic Mail, Journal of Network and System Management, 4 (3), 1996, pp. 279-297.

[8] Markowitch, O. Saeednia, S., Optimistic Fair Exchange with Transparent Signature Recovery, Proceedings of 5th International Conference Financial Cryptography 2001, LNCS, Springer-Verlag, 2001.

[9] Ray, I., and Ray, I., An Optimistic Fair Exchange E-commerce Protocol with Automated Dispute Resolution, Proceedings of 1st International Conference on E-Commerce and Web Technologies EC-Web 2000, LNCS, vol.1875, Springer-Verlag, Berlin, Germany, 2000, pp. 84-93.

[10] Schneier, B., Riordan, J., A Certified E-Mail Protocol, Proceedings of 13th Annual Computer Security Applications Conference, ACM Press, 1998, pp. 347-352.

[11] Shi, Q., Zhang, N., Merabti, M., Signature-based approach to fair document exchange, Communications, IEE Proceedings, 150 (1), 2003, pp. 21-27.

[12] Wu, C., Varadhrajan, V., Fair Exchange of Digital Signatures with Offline Trusted Third Party, International Conference on Information and Communication Security, 2001, pp. 466-470.

[13] Zhang, N., Shi, Q., Achieving Non-Repudiation of Receipt, The Computer Journal, 39 (10), 1996, pp. 844-853.

[14] Zhou, J., Deng, R., Bao, F., Some Remarks on a Fair Exchange Protocol, Proceedings of International Workshop on Practice and Theory in Public Key Cryptography, LNCS, vol. 1751, Springer-Verlag, 2000, pp. 46-57.