J. A. Onieva, J. Zhou, M. Carbonell, and J. Lopez, “Intermediary Non-Repudiation Protocols”, *5th Conference on Electronic Commerce*, pp. 207-214, 2003.

J. A. Onieva, J. Zhou, M. Carbonell, and J. Lopez , Intermediary Non-Repudiation Protocols , 5th Conference on Electronic Commerce , 207-214 , 2003

Intermediary Non-repudiation Protocols

Jose Antonio Onieva, Jianying Zhou
Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
onieva@i2r.a-star.edu.sg, jyzhou@i2r.a-star.edu.sg

Mildrey Carbonell, Javier Lopez
Computer Science Department, E.T.S. Ingenieria Informatica
University of Malaga, Spain, 29071 - Malaga
mildrey@crypto.lcc.uma.es, jlm@lcc.uma.es

Abstract

In commercial transactions, an intermediary might be involved to help transacting parties to conduct their business. Nevertheless, the intermediary may not be fully trusted. In this paper, we introduce the concept of intermediary (or agent) in a non-repudiation protocol, define the aims of intermediary non-repudiation protocols, and analyze their security requirements. We present a simple scenario with only one recipient, followed by a more complicated framework where multiple recipients are involved and collusion between them is possible.

1. Introduction

Electronic commerce helps businesses to expand their strategy and market, and for that, most of them are being shifted to the Internet or taking advantages of other digital sources. As the number and diversity of e-commerce participants grows, the complexity of purchasing (supplying, exchanging, ...) from a vast and dynamic array of goods and services needs to be hidden from end users. Collecting, verifying and storing evidence about the transactions are required, but might be undesirable for final entities when these transactions are undertaken with multiple entities and the volume is considerable. Hence, *intermediary* (IN) entities are useful in such scenarios to help final entities to carry out their business transactions. In addition, these entities can act as ‘hubs’, increasing the market and opportunities not only for customers but also for merchants.

Non-repudiation is an important requirement in electronic transactions [12]. It must not be possible for a merchant to claim that he sent the electronic goods when he did not. In the same way, it must not be possible for a customer to falsely deny having received the goods. Evidence should be collected to resolve these disputes arisen between participating entities in an electronic commerce scenario. Digital signature serves as a major type of cryptographic evidence, which links a message with its originator and also maintains the integrity of the message.

Fairness is also a desirable requirement in electronic transactions. A number of solutions to fair non-repudiation have been developed [7]. Some of them use a *Trusted Third Party* (TTP) that plays the role of a trusted intermediary between the participating entities. The major disadvantage of this approach is the communication bottleneck created at the TTP. Nevertheless, Zhou and Gollmann presented a protocol [13] where the TTP intervenes during each execution as a “low weight notary” rather than as an intermediary. Other solutions use an off-line TTP, assuming that participating entities have no malicious intentions and the TTP need not be involved unless there is an error in the protocol execution. This is called the optimistic approach. There are also solutions that eliminate the TTP’s involvement, but based on a strong requirement: all participating parties must have the same computational power. Therefore, in typical non-repudiation protocols three types of entities can be found: originators, recipients, and TTPs.

The research towards a generalization of non-repudiation, where multiple entities may participate in non-repudiation protocols, has been undertaken by Kremer and Markowitch [6, 9]. An extension that allows one originator to send different messages to many recipients in a general non-repudiation protocol appeared in [11]. Some work about multi-party scenarios in a related topic such as *fair exchange*, where entities have to exchange (accorded) items between them without loss of fairness, also exists [2, 3, 5].

The use of an intermediary to improve electronic transactions is not novel and can be found in [4, 10]. Nevertheless, no intermediary non-repudiation protocol exists to the best of our knowledge. Although two-party non-repudiation protocols could be used to implement an intermediary non-repudiation protocol, we will propose a new approach to improve the efficiency of such an implementation. In our new approach, a distrusted intermediary entity (different from the TTP) is introduced to facilitate the collection, verification, and storage of evidence on behalf of the originator. We demonstrate that the use of such an intermediary entity satisfies the security requirements expected in an e-commerce transaction.

The remainder of the paper is organized as follows. In Section 2, we define our model with a new entity involvement, identify the security requirements, and present an intuitive solution which will be compared later with our new approach. In Section 3, we present a simple intermediary non-repudiation protocol with one recipient only. In Section 4, we augment this scenario to the one where multiple recipients are involved. In Section 5, we further extend the scenario to a multi-recipient intermediary non-repudiation protocol for exchange of different messages. We conclude the paper in Section 6.

The following basic notation is used throughout the paper.

- $A \rightarrow B : X$: entity A sends message X to entity B
- $A \leftrightarrow B : X$: A fetches message X from B
- $A \Rightarrow \prod : X$: A multicasts message X to a set $\prod$
- X, Y : concatenation of messages X and Y
- u_P : the public key of user P
- $S_P(X)$: digital signature of user P over message X
- $E_K(X)$: encryption of message X with key K
- $h(X)$: one-way hash function with input X
- f : a flag indicating the purpose of a message

2. Model and Requirements

In [8], an agent-based commerce system ABECOS is proposed to achieve non-repudiation over electronic transactions. In this system, three principal entities are identified: a buyer, a seller, and a directory agent. The directory agent keeps information about other entities and acts as an intermediary broker that helps an entity to find other entities or agents that possess certain required capabilities. In this scenario, the intermediary agent is not involved in the non-repudiation protocol (see Figure 1).

2.1. The Model

We could extend the intermediary agent's role for non-repudiation purposes, thus liberating the originator of the non-repudiation protocol a part of the computation load to obtain evidence. In our system, an evidence database is maintained by the intermediary entity to store securely the evidence for each transaction. Depending on the applications, the evidence records may have an expiry date (and then, the dispute would not be settled after this date). Our system is flexible, and if originators request, evidence can be transferred to them during the protocol run (or even afterwards). The security policy defines who assumes which responsibility. This framework can also be extended to multiple recipients taking advantage of an intermediary acting as a hub. Figure 2 shows the model for which our protocol is designed.

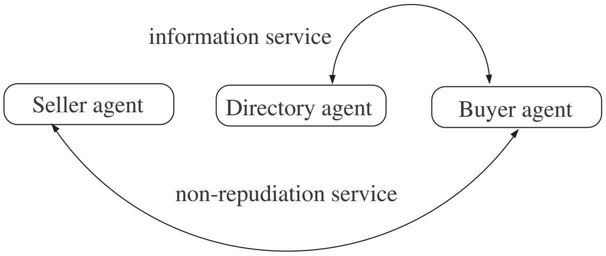

Figure 1. E-commerce scenario

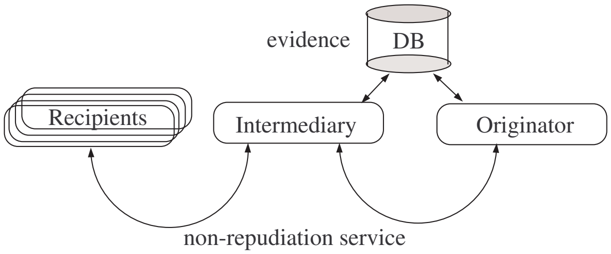

Figure 2. E-commerce scenario with an active intermediary agent

As we will see, fairness of a non-repudiation protocol depends overall on the behavior of this intermediary. The behavior of such an entity is usually related to its interests in the e-commerce scenario. Thus we can suppose this entity wishes to establish business relations with the participants and earn more profits by providing satisfactory services. Even so, we still do not treat the intermediary as a fully trusted entity in our protocol. Evidence of transactions carried out with this entity will be collected by the originator and recipients.

2.2. Security Requirements

An important requirement of non-repudiation services is *fairness* with which neither party can gain an advantage by quitting prematurely or otherwise misbehaving during a protocol. In other words, either all of honest participating entities obtain all the messages and the evidence needed, or none of them obtains items expected (i.e., the messages for the recipients and evidence of receipt for the originators). Another desirable requirement is *timeliness*, that is, all involved and honest parties are able to bring a protocol run to end without breach of fairness. *Confidentiality* might also be required. Only the intended parties might be able to disclose the message transmitted.

Evidence is essential to support non-repudiation services. In typical two-party non-repudiation protocols, at least two types of evidence must be collected by the participating entities.

- **Non-repudiation of origin:** It is intended to protect against the originator's false denial of having originated the message. Evidence of origin is generated by the originator or a trusted third party on its behalf, and will be held by the recipient.
- **Non-repudiation of receipt:** It is intended to protect against the recipient's false denial of having received the message. Evidence of receipt is generated by the recipient or a trusted third party on its behalf, and will be held by the originator.

In our model, an intermediary agent is involved in non-repudiation services, and plays not only the role of originator but also the role of recipient. New types of evidence are introduced, but they play the same function as the ones described above.

2.3. The First Solution

An intuitive solution to our intermediary non-repudiation model is to use two-party non-repudiation protocols. It can be described as a three-step scenario, where the originator first runs a fair non-repudiation protocol with the IN, and the latter with the recipients, then the last step for collecting final evidence between the IN and the originator (see Figure 3). If the fair non-repudiation protocol of [13] is used for the first and third steps, and its extended version for the exchange of multiple different messages [11] is used for the second step, at least 17 messages are needed to complete a transaction, without considering the number of encryption operations and digital signatures.

With the intuitive solution, a fair non-repudiation protocol is executed at Step 1 in order to make the intermediary be compromised to commitments of the messages destined to the recipients (5 messages). As a result, the IN obtains evidence of origin about the transaction requested by the originator, and the latter obtains the promise of the IN to do his best to deliver exactly those messages to the recipients indicated by the originator.

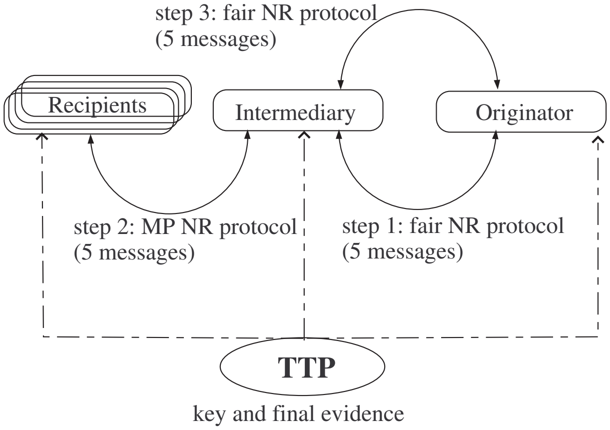

Figure 3. An intuitive solution to non-repudiation

Then, a multi-party non-repudiation protocol is used at Step 2 in order to compromise those recipients who reply to the commitments (5 messages). As a result, the recipients receive the service provided by the IN while the IN obtains the recipients' confirmation of the service.

Again, as in Step 1, an exchange is carried out between the originator and the IN about the result of the requested service at Step 3 (5 messages). This step permits the originator to obtain evidence about the result, and the IN to obtain the originator's agreement about the result.

At least 2 messages more are needed to complete the transaction. The originator lodges the keys of the commitments with the TTP, and all the entities collect the keys and final evidence from the TTP. It is out of the scope of this paper for a complete analysis of this solution. Further study of the basic protocols that compose this solution is encouraged.

3. A Simple Intermediary Non-repudiation Protocol

In this section, we present our new approach for intermediary non-repudiation by first introducing a simple intermediary non-repudiation protocol with an IN entity and only one recipient. In a gradual manner we will extend this approach, reaching a framework with multiple recipients and multiple messages in the following sections.

3.1. A Simple Protocol

As we noted in Section 2, the intermediary plays a critical role in this scenario, so it is important to analyze its behavior. If the IN has any interest (any charge with the originator or the recipients) in a transaction, it will be willing to reach a successful transaction. But occasionally, the IN may collude with another (external or internal) entity and, for instance, hide some evidence. Therefore, we assume the intermediary is not fully trusted.

Here we presume that the IN, which could be selected by the originator, is not going to hide the initial messages from the originator to the intended parties. (In Section 4.4 we will explain how to get rid of this assumption.) The simplest approach comes when the originator wishes to send a message to one recipient. In this scenario, the IN entity does not play the role of a hub. Nevertheless, it communicates directly with the recipient, and could help the originator not only in the non-repudiation protocol itself but also in the preliminary steps, such as search of a recipient and a product, price agreement, etc. For this purpose, we introduce a new term *request*, that gives the IN some information about the service to be provided. The following notation is used in the protocol description.

- $All = O, IN, R$: All the entities excluding the TTP
- M: message being sent from the originator O to the recipient R
- k: key being selected by O
- $c = E_k(M)$: encrypted message for R with key k
- $l = h(M, k)$: label of message M
- t: a timeout chosen by O, before which the TTP has to publish some information
- $EOOc = S_O(feoo, IN, R, l, t, h(request), h(c))$: evidence of origin of c generated by O
- $EOOI = S_{IN}(feooi, R, O, l, t, c)$: evidence of origin of c issued by the IN for R
- $EORc = S_R(feor, IN, O, l, t, c, u_R)$: evidence of receipt of c generated by R
- $EORI = S_{IN}(feori, O, R, l, t, h(request), h(c), u_R)$: evidence of receipt of c issued by the IN for O
- $Sub_k = S_O(fsub, TTP, IN, R, t, l, E_{u_R}(k), u_R, EORI)$: evidence of submission of the key to the TTP generated by O
- $Con_k = S_{TTP}(fcon, All, l, t, E_{u_R}(k), u_R, EORI)$: evidence of confirmation of the key issued by the TTP

The protocol is as follows.

1. $O \rightarrow IN \quad : feoo, IN, R, l, t, request, c, EOOc$
2. $IN \rightarrow R \quad : feooi, R, O, l, t, c, EOOI$
3. $R \rightarrow IN \quad : feor, IN, O, l, u_R, EORc$
4. $IN \rightarrow O \quad : feori, O, R, l, u_R, EORI$
5. $O \rightarrow TTP \quad : fsub, TTP, IN, R, t, l, E_{u_R}(k), u_R, EORI, h(request), h(c), Sub_k$
6. $All \leftrightarrow TTP : fcon, TTP, All, l, E_{u_R}(k), EORI, Con_k$

The protocol works in the following way.

Step 1: O sends to the IN the request information and evidence of origin corresponding to the encrypted message c. If confidentiality for the request information is needed, an encryption operation with the IN's public key can be performed. The encrypted message c may be some sensitive information, for instance bank account data, that O is not intended to reveal to the IN. There is no breach of fairness if the protocol stops.

Step 2: The IN distributes O's information (maybe after a negotiation or agreement with R), and sends R evidence of involvement in the transaction $EOOI$, such that if IN try to change any string, it will be detected in a dispute phase. Again, fairness is maintained if the protocol stops.

Step 3: R replies with evidence of receipt of encrypted message c. R's public encryption key u_R is included in $EORc$ to make it undeniable when O uses it at Step 5 to distribute key k. In this way, the originator does not need to verify or retrieve any public key information about the recipient. The protocol still remains fair if it stops.

Step 4: The IN replies to O, indicating that R agreed the transaction. At the same time evidence of receiving *request* and c is given to O. O will check this evidence carefully before proceeding to the next step, since this is the only evidence O will collect from the IN and will be used by O in case of disputes to prove the IN's responsibility of the exchange. The IN will store R's evidence of receipt in the Evidence Database, and O can retrieve it later if needed. The IN cannot claim that it didn't store this evidence since $EORI$ demonstrates it did if a dispute arises. No party is benefited if the protocol stops at this step.

Step 5: O submits the key encrypted with R's public key to the TTP, such that only the intended recipient will be able to disclose the message. This submission should be carried out before deadline t. Before going to the next step, the TTP will check that a correct signature of the IN on $EORI$ is embedded into the message. It helps the IN to demonstrate that O has been notified about the delivery result in case a dispute arises.

Step 6: The TTP releases the encrypted key. O fetches Con_k as evidence that it submitted the key in time to complete the transaction. The IN fetches Con_k as evidence that O accepted $EORI$ and thus the service offered by the IN. R obtains the key to decrypt c and fetches Con_k as evidence to prove its origin. $EORI$ is included in this message to permit R to verify the signature Con_k.

At the end of the protocol, each party will hold the corresponding evidence.

- The originator should collect $EORI$ and Con_k as evidence of receipt.
- The IN should collect $EOOc$, $EORc$, and Con_k as evidence of origin and evidence of receipt, respectively, which allows the IN to demonstrate its good behavior during the protocol.
- The recipient should collect $EOOI$ as evidence of origin of c issued by the IN. Con_k must also be collected as evidence of origin of the key.

Our protocol takes only 6 steps, improving the first intuitive solution we presented in Section 2.3 on the number of messages sent over the network. In our protocol, *anonymity* could be preserved, that is, unless the originator is willing to communicate with a pre-selected recipient, neither the originator nor the recipient needs any knowledge (i.e. digital certificates) about each other in order to reach a successful protocol end. As we can see above, only the IN needs final entities' information (i.e. digital certificates) in order to verify digital signatures while the final entities only need the IN's digital certificate during the protocol execution.

3.2. Dispute Resolution

In our model, disputes might arise between any pair of three parties. If the evidence has an expiry date, the disputes should be settled before this date.

Disputes between Originator and Recipient

If O denies sending message M, R shows the arbitrator evidence $EOOc$ and Con_k. With $EOOc$, the arbitrator checks whether O originated c. With Con_k, the arbitrator checks whether $E_{u_R}(k)$ is encrypted with u_R and published by O via the TTP. If both checks are positive, the arbitrator settles that message M is from O. In order to obtain $EOOc$, the recipient must retrieve this evidence from the Evidence Database. But if the IN precludes the recipient's access to the evidence (or it is not valid), the recipient should present $EOOI$ to the arbitrator, proving that it received the encrypted message c from the IN and now the responsibility of submitting $EOOc$ lies on the latter.

If R denies that O published key k, O shows the arbitrator evidence Con_k[1]. With Con_k, the arbitrator checks whether u_R is R's public encryption key. (O may further retrieve $EORc$ from the IN to support it.) Besides, the arbitrator checks whether $E_{u_R}(k)$ is encrypted with u_R and became available by the predefined time t. If both checks are positive, the arbitrator settles that O published key k.

Although it is not strictly necessary, if the originator or the recipient collaborates with the IN in the dispute resolution and retrieve evidence from the Evidence Database, the IN can disclose the identity (i.e. digital certificates) of the participants in such a case.

Disputes between Originator and Intermediary

If the IN denies having received any request labeled l from O, O presents $EORI$ and the arbitrator checks the IN's signature on it as well as validity of the label. If successful, the arbitrator settles that O sent the request to the IN. If O denies having received a response from the IN for a labeled transaction l, the IN presents Con_k and the arbitrator checks the TTP's signature on it. If successful, the arbitrator settles that O published the key due to receipt of a response from the IN.

Disputes between Recipient and Intermediary

If the IN denies delivering message c to R, R presents evidence $EOOI$ and the arbitrator checks the IN's signature on it. If successful, the arbitrator settles that c, originated from O, is delivered by the IN to R. If R denies having received message c, the IN presents $EORc$ and the arbitrator checks R's signature on it. If successful, the arbitrator settles that the IN delivered c to R.

4. Extension to Multiple Recipients

The intervention of an IN entity becomes more interesting when multiple parties can be involved in a transaction and then the IN entity acts as a hub. That is, the originator sends information to an intermediary about the transaction, and the latter broadcasts this transaction among multiple recipients (i.e., depending on the information contained in *request*). A new protocol that combines the intermediary protocol presented in Section 3.1 and the multi-party non-repudiation protocol [6] is introduced in this section.

In order to release the key only to the honest parties, a group encryption mechanism is needed. We use [1] that allows the encryption of a message to be decrypted by an intended group of recipients using any public-key encryption scheme.

4.1. A Multi-recipient Protocol

Some useful new notation in the protocol description is as follows.

- $All = O, IN, R'$: All the entities excluding the TTP
- R: set of intended recipients
- R': set of recipients that replied to the intermediary with the evidence of receipt
- $u_{R'}$: set of public encryption keys belonging to R'

---

[1] Note that the arbitrator has just to verify Con_k, since $EORI$ was checked by TTP before publishing step 6

- R_i: Each of the intended recipients with $i \in \{1..|R|\}$
- $E_{R'}(k)$: a group encryption scheme that encrypts k for the group R'
- $EOOI = S_{IN}(feooi, R, O, l, t, EOOc)$: evidence of origin of c issued by the IN for R
- $EORc_i = S_{R_i}(feor, IN, O, l, t, h(c), u_{R_i})$: evidence of receipt of c generated by R_i
- $EORI = S_{IN}(feori, O, R', l, t, h(request), h(c), u_{R'})$: evidence of receipt of c issued by the IN for O containing the identities of the recipients who replied
- $Sub_k = S_O(fsub, TTP, IN, R', l, t, E_{R'}(k), u_{R'}, EORI)$: evidence of submission of the key to the TTP generated by O
- $Con_k = S_{TTP}(fcon, All, l, t, E_{R'}(k), u_{R'}, EORI)$: evidence of confirmation of the key issued by the TTP

The protocol is as follows.

1. $O \rightarrow IN$: $feoo, IN, R, l, t, request, c, EOOc$
2. $IN \Rightarrow R$: $feooi, R, O, l, t, h(request), c, EOOc, EOOI$
3. $R_i \rightarrow IN$: $feor, IN, O, l, u_{R_i}, EORc_i$
4. $IN \rightarrow O$: $feori, O, R', l, u_{R'}, EORI$
5. $O \rightarrow TTP$: $fsub, TTP, IN, R', t, l, E_{R'}(k), u_{R'}, EORI, h(request), h(c), Sub_k$
6. $All \leftrightarrow TTP$: $fcon, TTP, All, l, E_{R'}(k), EORI, Con_k$

Minor changes are introduced to the previous protocol. In this situation, the IN should store all the evidence collected from the honest recipients in the Evidence Database. In case of disputes, the resolution process remains unchanged (see Section 3.2). Although the recipients receive evidence of origin $EOOc$ from O, we assume that they do not store this evidence, even though they may. We explain why they need this evidence in Section 4.4. Public encryption keys are included by each recipient to make them undeniable when O uses them at the group encryption scheme to distribute key k. In this way, the originator does not need to verify or retrieve any public key information about recipients. Besides, each recipient only needs to verify that its own public encryption key signed in Con_k is valid.

In this scenario, we should avoid the IN sending a $R'' \neq R'$ to O. If $R'' \supset R'$, then the intermediary claims that some recipients replied when they did not. Some solutions exist depending on the transaction's type. If the disclosure of the message can be brought back or the transaction can be cancelled after a dispute resolution, O can request to settle the dispute and the IN will not be able to present all the evidence of receipt. If it is not possible (i.e., for more critical transactions or exchanges), the IN should send all the evidence of receipt to O at Step 4. But O only needs to keep $EORI$, and may not maintain the evidence of receipt generated by each recipient after verifying that $R'' = R'$. O will terminate the protocol run if $R'' \neq R'$.

If $R'' \subset R'$ then the intermediary hides some evidence of receipt from some of the honest recipients. Here, the solution requires a recovery sub-protocol (see Section 4.3) which permits these honest entities communicate directly to the TTP about their commitment to the transaction.

4.2. Group Encryption

A group encryption scheme is used to encrypt the key *k* for the recipients R' in our protocol. It is based on a public-key encryption scheme and on the Chinese remainder theorem. This method is generic as it can use any public-key cryptosystem. Let us instantiate it for our protocol.

- Let u_{R_i} and v_{R_i} be the public and private keys of R_i, respectively (R_i corresponds to all parties that belong to R').
- Each recipient of R' receives a random integer $P_i < E_{u_{R_i}}(k)$ such that all P_i are pair-wise relatively prime. (When choosing randomly large primes or multiplications of distinct primes for example, the probability of obtaining two numbers that are not relatively primes is negligible.)
- O computes $X \equiv E_{u_{R_i}}(k)\ mod\ P_i$. As all of P_i are prime integers, using the Chinese remainder theorem, only one solution is obtained from this equation. Hence, $E_{R'}(k) \equiv X$. Each recipient R_i can obtain *k* by computing $X \equiv E_{u_{R_i}}(k)\ mod\ P_i$ using her private key v_{R_i}.

In our protocol, O is required to include P_i in Sub_k to make the encryption of k undeniable. (For simplicity, it is omitted in the above protocol.)

4.3. Recovery Sub-protocol

Let $t1 < t$ be a deadline time after which the recovery protocol cannot be launched by any recipient. The previous notation is modified as follows.

- $EOOc = S_O(feoo, IN, R, l, t, \mathbf{t1}, h(request), h(c))$: evidence of origin of c generated by O
- $EOOI = S_{IN}(feooi, R, O, l, t, \mathbf{t1}, EOOc)$: evidence of origin of c issued by the IN for R
- $EORc_i = S_{R_i}(feor, IN, O, l, t, \mathbf{t1}, h(c), u_{R_i})$: evidence of receipt of c generated by R_i
- $\mathbf{EORER_i} = S_{IN}(feorer, l, EORc_i)$: evidence of receipt of evidence of receipt of c from R_i issued by the IN

The main protocol is modified as well.

1. $O \rightarrow IN$: $feoo, IN, R, l, t, \mathbf{t1}, request, c, EOOc$
2. $IN \Rightarrow R$: $feooi, R, O, l, t, \mathbf{t1}, h(request), c, EOOc, EOOI$

3. $R_i \rightarrow IN \quad : feor, IN, O, l, u_{R_i}, EORc_i$
4. $\mathbf{IN \rightarrow R_i} \quad : \mathbf{feorer, l, EORER_i}$
5. $IN \rightarrow O \quad : feori, O, R', l, u_{R'}, EORI$
6. $O \rightarrow TTP \quad : fsub, TTP, IN, R', t, l, E_{R'}(k), u_{R'}, EORI, h(request), h(c), Sub_k$
7. $All \leftrightarrow TTP : fcon, TTP, All, l, E_{R'}(k), u_{R'}, EORI, Con_k$

The recovery sub-protocol is as follows.

5.a. $R_i \rightarrow TTP : \; frec, TTP, IN, O, R, l, t, t1, EOOc, u_{R_i}, h(c), h(request), EORc_i$
If (If t1 is correct) : TTP ignores message
Else {
5.b. $TTP \rightarrow O : frec, O, R_i, l, u_{R_i}, EORc_i$
5.c. O adds R_i into R'}

The recovery sub-protocol will be launched only in case of the IN's misbehavior or channel failure. A new step has been introduced in the main protocol, such that the IN must reply to every evidence of receipt ($EORc_i$). If R_i receives $EORER_i$ at Step 4 but the IN does not include him in R', the recipient can present $EORER_i$ to the adjudicator in a dispute resolution.

If R_i does not receive $EORER_i$ at Step 4, considerably before $t1$, the recipient should launch the recovery sub-protocol to contact the TTP directly. The TTP checks that the message $5.a$ arrives before $t1$ and that the same t1 is signed by O on $EOOc$. Then the TTP sends the recovery information to O and the latter will include R_i into R' for the group encryption of key k after validating the evidence and checking that R_i belongs to R.

R_i may launch the recovery sub-protocol (several times) even when the IN behaves honestly. However, this does not give the recipient any benefit. On the contrary, the recipient may need to pay more when requesting this service from the TTP.

4.4. Collaboration among Recipients

A problem might arise if the intermediary entity sends the messages to the recipients in a selective manner. The IN can always claim that some recipients did not reply.

Usually, the IN would not misbehave in such a way if it has any interest in bringing a transaction to its end. Nevertheless, the IN may collude with another internal or external entities and exclude some recipients from the transaction if it can get more benefits. In this case, the recipients should collaborate in order not to be excluded. After receiving Step 2 of the protocol, each recipient that did not receive this step *again* can distribute this message to the rest of recipients. Otherwise, it just continues. In order to obtain the group R of recipients before distributing any message, the recipient R_i should verify that the group R sent by the IN matches with the one included in $EOOc$, thus, loosing O's anonymity.

At least one honest entity should receive Step 2 to avoid the IN's misbehavior. The collaboration among recipients will be recommendable depending on the transaction type and the network latency since this solution needs more messages.

5. Further Extension to Multiple Messages

Frequently, in e-commerce applications, the originator needs to send different messages to recipients in the same transaction. A modification can be made to distribute different messages to the intended parties. The originator may send these different messages and instructions on how to split them for each recipient in the *request* information transmitted in the first message.

In this extension, the use of the same key for all users creates a new problem. As messages are different, when the same key is used for encryption, and after the key *k* is published, any recipient will be able to read the messages destined to the other recipients (by eavesdropping the messages that are transmitted between the IN and R). This problem can be solved with the technique proposed in [11].

Let R be a group of n recipients and M_i the different plain messages that the IN sends to each R_i, with $i \in \{1..n\}$. The following notation is used in the protocol description.

- n_i: a random value generated by O for each R_i
- $x_i = E_{u_{R_i}}(n_i)$: encryption of n_i with R_i's public key
- $k_i = k \; xor \; n_i$: a key for each R_i
- $c_i = E_{k_i}(M_i)$: encrypted message with a key k_i for each R_i
- $l_i = h(M_i, k_i)$: label of message M_i
- L' : concatenation of labels from recipients who replied
- $EOOc_i = S_O(feoo, IN, R_i, l_i, x_i, u_{R_i}, t, h(request), h(c_i))$: evidence of origin of c_i generated by O
- $C' = l_1 c_1 x_1 u_{R_1} EOOc_1 ... l_n c_n x_n u_{R_n} EOOc_n$: concatenation of label, encrypted message, encrypted random number, public encryption key, and evidence of origin for each recipient
- $EOOI_i = S_{IN}(feooi, R_i, O, l_i, x_i, u_{R_i}, t, h(c_i))$: evidence of origin of c_i issued by the IN for R_i
- $EORc_i = S_{R_i}(feor, IN, O, l_i, x_i, u_{R_i}, t, c_i)$: evidence of receipt of R_i
- $EORI = S_{IN}(feori, O, R', L', t, h(request), h(C'))$: evidence of receipt of C' issued by the IN for O
- $Sub_k = S_O(fsub, TTP, IN, R', L', t, E_{R'}(k), EORI)$: evidence of submission of the key to the TTP generated by O

- $Con_k = S_{TTP}(fcon, All, L', t, E_{R'}(k), EORI)$: evidence of confirmation of the key issued by the TTP

The protocol is as follows.

1. $O \rightarrow IN$: $feoo, IN, R, t, request, C'$
2. $IN \rightarrow R_i$: $feooi, R_i, O, t, l_i, c_i, x_i, u_{R_i}, EOOI_i$
3. $R_i \rightarrow IN$: $feor, IN, O, l_i, EORc_i$
4. $IN \rightarrow O$: $feori, O, R', L', EORI$
5. $O \rightarrow TTP$: $fsub, TTP, IN, R', L', t, E_{R'}(k), EORI, h(request), h(C'), Sub_k$
6. $All \leftrightarrow TTP$: $fcon, TTP, All, L', E_{R'}(k), EORI, Con_k$

The originator selects the intended public keys that are going to be used in the encryption of n_i. If the recipient disagrees (i.e., its digital certificate has expired or been revoked), it should stop the protocol after receiving Step 2.

This protocol has the same properties as the one in the previous section. However, if no trust is deposited on the IN, some external mechanism should be found to ensure this entity will distribute all the messages to the intended parties. (As each recipient will not receive the same message, no collaboration is possible among the recipients to solve this problem as proposed in section 4.4.)

6. Conclusion

In e-commerce, where business cannot be conducted face to face, it is not realistic to expect all parties to trust and to cooperate with one another during the entire purchasing process. Since various participants, all having different requirements, operating in different, distributed and heterogeneous environments, are encompassed in an e-commerce interaction, non-repudiation is identified as a key requirement for designing transaction models and protocols.

As more and more digital goods and services appear on the Internet, the end-user will not only find more business opportunities but also find more difficult to make a transaction, especially when multiple recipients are involved. For that, security issues out of the scope of e-commerce transactions should be hidden from end-users.

In this paper, we analyzed a new entity that takes part in the non-repudiation protocol. This entity can be just another module in an agent-based system, facilitating the originator to carry out an e-commerce transaction. We introduced different scenarios such that our approach can be easily fitted into each of them. We demonstrated the advantages for end-users in the use of an intermediary on reducing the evidence storage requirements and gathering different recipients. The originator can be kept anonymous to the recipients, and vice versa, in this intermediary non-repudiation protocol as long as the originator and the recipients do not need to verify each other's evidence. The intermediary entity can be distrusted and our approach maintains the security requirements for a non-repudiable e-commerce transaction.

References

[1] G. Chiou and W. Chen. Secure broadcasting using the secure lock. *IEEE Transaction on Software Engineering*, 15(8), August 1989.

[2] M. Franklin and G. Tsudik. Secure group barter: Multi-party fair exchange with semi-trusted neutral parties. In *Proceedings of Financial Cryptography 1998*, volume 1465 of *Lecture Notes in Computer Science*, pages 90–102. Springer, February 1998.

[3] N. González-Deleito and O. Markowitch. An optimistic multi-party fair exchange protocol with reduced trust requirements. In *Proceedings of 4th International Conference on Information Security and Cryptology*, volume 2288 of *Lecture Notes in Computer Science*, pages 258–267, Seoul, Korea, December 2001. Springer.

[4] S. Ketchel and H. Garcia-Molina. Distributed commerce transactions, 1997.

[5] J. Kim and J. Ryou. Multi-party fair exchange protocol using ring architecture model. In *Proceedings of Japan-Korea Joint Workshop on Information Security and Cryptology*, January 2000.

[6] S. Kremer and O. Markowitch. A multi-party non-repudiation protocol. In *Proceedings of 15th IFIP International Information Security Conference*, pages 271–280, Beijing, China, August 2000.

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

[8] C.-C. Liew, W.-K. Ng, E.-P. Lim, B.-S. Tan, and K.-L. Ong. Non-repudiation in an agent-based electronic commerce system. In *Proceedings of 1999 DEXA International Workshop on Electronic Commerce and Security*, pages 864–868, Florence, Italy, September 1999.

[9] O. Markowitch and S. Kremer. A multi-party optimistic non-repudiation protocol. In *Proceedings of 3rd International Conference on Information Security and Cryptology*, volume 2015 of *Lecture Notes in Computer Science*, pages 109–122, Seoul, Korea, December 2000. Springer.

[10] T. Mullen and M. Wellman. The auction manager: Market middleware for large-scale electronic commerce. In *Proceedings of 3rd USENIX Workshop on Electronic Commerce*, pages 37–48, Boston, Massachusetts, September 1998.

[11] J. Onieva, J. Zhou, M. Carbonell, and J. Lopez. A multi-party non-repudiation protocol for exchange of different messages. In *Proceedings of 18th IFIP International Information Security Conference*, Athens, Greece, May 2003.

[12] J. Zhou. *Non-repudiation in electronic commerce*. Computer Security Series. Artech House, 2001.

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