Selective Receipt in Certified E-Mail

Steve Kremer and Olivier Markowitch
{skremer,omarkow}@ulb.ac.be

Université Libre de Bruxelles
Bd du Triomphe C.P. 212, 1050 Brussels, Belgium

Abstract. Traditional pen and paper transactions are becoming more and more replaced by equivalent electronic services. Therefore electronic e-mail should also provide enhanced services as those provided by traditional mail. In this paper we present new optimistic protocols for certified e-mail. The major contribution of our paper is the definition of a new property, specific to certified e-mail: *no author-based selective receipt*. This property requires that once the identity of the author of the mail is known, the receipt can not be refused any more. We present two certified e-mail protocols respecting this property.

1 Introduction

Due to the tremendous growth of open network in general and the Internet in particular, traditional 'pen and paper' transactions are becoming more and more replaced by equivalent electronic services. Probably the best example to illustrate this fact is electronic mail. Millions of e-mails are sent every day over the Internet. However the traditional post offers more sophisticated services, such as express mail, compensation for lost mail or certified mail. Hence, the electronic equivalent should offer similar services. Express mail deals with a fast delivery of mails before a fixed deadline. On networks, fast delivery is part of the nature of electronic mails and is naturally provided, unless at least one link on each route between the sender and the recipient is broken. However, the Internet is designed to resist a high number of failures. In a similar way, compensation of lost mails is almost never necessary when using e-mail: an electronic item does not physically exist and cannot be lost in the same way as a traditional letter. Although e-mails can be mis-routed or lost due to network failures, the loss is insignificant as multiple copies of an item can easily be created. If the mail has to be delivered before a given deadline, the loss of a message, or a permanent network failure can be a problem. In that case, compensation could be an expected property, but this paper does not focus on this issue. The third service we mentioned is certified mail. This service provides a proof of delivery to the sender. In opposition to express mail and compensation for lost items, the electronic version of certified delivery is more complicated to provide than the traditional one. When we send a traditional certified mail, a person assures the exchange of the mail against a receipt. The mail is only handed to the receiver, after he has signed the receipt. On the other hand the receiver is sure that once he signed the receipt he will receive the mail, as the postman, holding the mail, stands in front of him. In an electronic environment, guaranteeing a certified mail is more complicated. Imagine the following scenario where Alice wants to send a mail to Bob: she sends an e-mail to Bob who is expected to return a receipt. In this scenario, Bob can first read the mail, and then decide whether he wants to admit having received it or not. He can choose not to send the receipt although he read the mail. If we change the scenario such that Bob sends the receipt before having received the message, we face a different problem: Alice could get the receipt and claim that Bob received a message he never saw. In order to overcome these problems we have to use specialized protocols.

Before looking at existing certified e-mail protocols, we first discuss some related topics: fair exchange [1, 2, 11] and non-repudiation [8, 9, 14, 16] protocols. All of these topics have the same underlying problem: an exchange of secrets in which none of the entities gains a significant advantage over the other one. In a fair exchange protocol an entity wants to exchange one or several items against one or several other items in a fair way, i.e. in a way that at the end of the protocol either both got their expected items or none of them got any valuable information. A certified e-mail protocol can be seen as an instance of a fair exchange protocol. However, in fair exchange protocols the expected item, or a description of it, is generally known a priori, before the exchange takes place. In a certified e-mail protocol, the message that is sent is not known to the receiver. In a non-repudiation protocol, an entity, Alice, sends a message to another entity Bob. At the end of the protocol, Alice is expected to have a non-repudiation of receipt evidence, i.e. an evidence that Bob received the message, and Bob is expected to have a non-repudiation of origin evidence, i.e. an evidence that Alice is the author of the message. Those evidences can, in case of dispute be presented to a judge. The difference between certified e-mail and non-repudiation protocols is not very clear in literature and will be discussed later in more detail.

First solutions to these exchange problem were based on gradual exchange protocols [13]. The disadvantages of this approach are the requirement of equivalent computational power and the network overhead. The second approach is a probabilistic one [4, 9]. Generally, the probability to cheat the other entity can be decreased by increasing the number of messages necessary in the protocol. To avoid the communication overhead, a different approach using a trusted third party (TTP) has been introduced. Both entities can send their items to the TTP that forwards them to the respective entities. However, this may create a communication and computation bottleneck at the TTP. To overcome this bottleneck, independently, Micali and Asokan et al. [1] introduced the optimistic approach in the context of fair exchange protocols. The rationale is that the TTP only intervenes in case of a problem, i.e. an entity is trying to cheat or a communication fails at a crucial moment. In an optimistic protocol, the TTP is said to be offline, while it is online in non-optimistic protocols. The optimistic approach has received most attention in recent literature.

In literature, there have been several papers addressing explicitly certified e-mail with either online TTP [7, 10, 15] or offline TTP [2, 3, 11]. However, all except Zhou et al. in [15] do not make a difference between a certified e-mail protocol and a non-repudiation protocol. In [15], Zhou et al. argue that non-repudiation of origin is not a required service—in fact, traditional certified mail does not provide it—and define a certified e-mail protocol to be a service that must be offered by an external delivery agent, in the same way it is realized by the post in traditional mail. We do not follow this last point, as the definition of a service provided by a protocol should not depend on *how* it is provided. Moreover, Zhou's approach does not permit to use an optimistic approach, which is, in most cases, more efficient. In this paper, we propose an optimistic certified e-mail protocol and we introduce an additional property, specific to certified e-mail, that has not been discussed previously: *author based selective receipt*. Generally, it is required that once the message is known to the recipient, he cannot prevent the protocol from delivering the receipt. It seems to us that this property is not sufficient: we believe that once the identity of the author is revealed, the receipt has to be delivered to the sender, and of course the message to the recipient. In traditional certified mail, one does not get any information about the mail, neither the content nor the origin, before having signed the receipt. To better understand the crucial importance of that property, consider the following real life example. If a person does not pay the rent of his flat, he can refuse a certified mail coming from his landlord. He does not need to read the letter to guess that the landlord is claiming the rent and wants a proof for this claim. Knowing the identity of the sender would reveal enough information to guess the content of the letter. If we suppose that a network address, e.g. an IP address is sufficient to identify a person we need to introduce an additional mechanism. We propose to use a third party (or several third parties) as an anonymity provider. Hence, our protocol is not entirely optimistic. However a third party providing anonymity is different from the TTP classically used in this kind of protocols. The third parties used for anonymizing communications do not have to generate evidences that need to be verified by an adjudicator during a dispute. An anonymity provider does not need to have a jurisdiction on evidence generation, as a TTP. Also, they do not need to be entirely trusted as they are observable by any exterior party, that could detect "strange" behaviors.

The rest of the paper will be structured as follows. In section 2, we will define the properties required by a certified e-mail protocol. In section 3, we present two variants of a new certified e-mail protocol: the first protocol does not provide data confidentiality, while the second one does. We go on discussing several solutions to the anonymity requirements of our protocols and finally conclude.

2 Properties

In this section we define all the properties a certified e-mail protocol is required to provide. We define them with respect to a sender, that we call Alice and a recipient we call Bob. Throughout he rest of this paper we also assume that no party acts against its own interests. This assumption is rather natural and discards scenarios, where a dishonest party, i.e. a party not following the protocol, could break some of the underneath defined properties by harming itself.

The first property is *non-repudiability.* Two kinds of non-repudiation services are required: non-repudiation of receipt and non-repudiation of origin.

Definition 1 (Non-repudiation of receipt). *A certified e-mail protocol provides non-repudiation of receipt, if and only if it generates a non-repudiation of receipt evidence, destined to Alice, that can be presented to an adjudicator, who can unambiguously decide whether Bob received a given message or not.*

Definition 2 (Non-repudiation of origin). *A certified e-mail protocol provides non-repudiation of origin, if and only if it generates a non-repudiation of origin evidence, destined to Bob, that can be presented to an adjudicator, who can unambiguously decide whether Alice is the author of a given message or not.*

Non-repudiation of origin is not necessary in a certified e-mail protocol, as it is not provided by a classic certified mail service. However, in most papers on certified e-mail it is treated as a mandatory property [3, 7, 10]. Moreover, in optimistic protocols a non-repudiation of origin evidence is provided quite naturally, as Bob must prove he received a message from Alice, when contacting the TTP. Both protocols that we present in the following section will provide non-repudiation of origin.

A second property the protocol must respect is fairness.

Definition 3 (Fairness). *A certified e-mail protocol is* fair *if and only if at the end of a protocol execution either Alice got the non-repudiation of receipt evidence, and Bob got the corresponding mail (as well as the non-repudiation of origin evidence if required), or none of them got any valuable information.*

Fairness ensures that none of the entities can cheat the other, i.e. arrives in a situation where either Alice or Bob has got his expected item, and the other has no mean of receiving his item anymore.

Another important property is timeliness.

Definition 4 (Timeliness). *A certified e-mail protocol provides* timeliness *if and only if all honest parties always have the ability to reach, in a finite amount of time, a point in the protocol where they can stop the protocol while preserving fairness.*

Timeliness assures that an entity does not need to keep open protocol runs for an infinite amount of time. Such a situation could occur if an entity is not sure whether the other entity stopped the protocol or not. It must always be possible for an entity to quit a protocol, without giving the possibility to the other entity to gain any advantage.

An optional property that can sometimes be required is confidentiality.

Definition 5 (Data confidentiality). *A certified e-mail protocol is said to provide* data confidentiality*, if and only if Alice and Bob are the only entities that can extract the content of the sent mail out of the protocol messages.*

This property ensures that no one can read the content of a sent mail, by for instance listening to the communication channels. In pen and paper transactions this property should be ensured by the envelope. Moreover, confidentiality is not always required. As adding confidentiality harms the efficiency of the protocol and is not always requested, we will present two protocols, where the first one does not provide confidentiality. It could also be interesting to anonymize Alice's identity from all entities exterior to the protocol and to provide *entity confidentiality*. In the remaining of this paper, we only consider data confidentiality.

The last property we require is *no selective receipt*. Selective receipt comes in two flavors.

Definition 6 (No message based selective receipt). *A certified e-mail protocol does not allow* message based selective receipt *if and only if once the message is known to Bob, he cannot prevent delivery of a receipt to Alice.*

No message based selective receipt is directly implied by the fairness requirement. We mention this property, as it has been discussed before in literature.

Definition 7 (No author based selective receipt). *A certified e-mail protocol does not allow* author based selective receipt *if and only if once the identity of the author is known to Bob, he cannot prevent delivery of a receipt to Alice.*

No author based selective receipt is a new property introduced in this paper. We believe that hiding the content of a mail, while stopping the protocol is still possible, is not sufficient. One should also hide the author's identity. In many cases the origin of a message leaks enough information to guess the mail's content[1].

3 The Protocols

In this section, we present two variants of a new protocol. The first protocol does not provide confidentiality, while the second one does. We believe that for efficiency reasons it is important that a user can decide to require confidentiality or not, as any additional property requires additional signatures or ciphers to be computed. We will discuss the security of each of the protocols with respect to the different properties. In both protocols, we assume that the communication

[1] Of course, *every* sent certified e-mail has to provide this property. Otherwise, a recipient always rejects anonymous messages.

channels between the TTP and respectively Alice and Bob are resilient, i.e. all data sent on such a channel arrive correctly after a finite, but unknown amount of time, although there may be delays. Channels between Alice and Bob may be unreliable, that means that data may accidentally be lost. Each of the protocols consists of three sub-protocols: a main protocol, an abort protocol and a recovery protocol. The main protocol performs the exchange. Both the abort protocol and the recovery protocol are used in case of problems to contact the TTP and either cancel the exchange or force a correct termination of the exchange.

3.1 A certified e-mail protocol

Notations We use the following notations to describe the protocol.

- $X \longrightarrow Y$: transmission from entity X to entity Y
- $X \xrightarrow{@} Y$: anonymous transmission from entity X to entity Y
- $X \xleftarrow{@} Y$: reply from entity Y to entity X on an anonymous channel
- $h()$: a collision resistant one-way hash function
- $E_k()$: a symmetric encryption function under key k
- $D_k()$: a symmetric decryption function under key k
- $E_X()$: a public-key encryption function under X's public key
- $S_X()$: the signature function of entity X
- m: the mail sent from A to B
- k: a fresh session key A uses to cipher m
- $c = E_k(m)$: the result of a symmetric ciphering of m under the key k
- $\ell = h(m, A, B, k)$: a label that, in conjunction with the identities (A,B), uniquely identifies a protocol run[2]
- f: a flag indicating the purpose of a message

The protocol generates the following evidences.

- $\mathsf{EOO} = S_A(f_{\mathsf{EOO}}, A, B, TTP, \ell, h(c), k)$
- $\mathsf{EOR}_c = S_B(f_{\mathsf{EOR}_c}, B, TTP, \ell, h(c), E_{TTP}(f_{\mathsf{EOO}}, A, B, \ell, k, \mathsf{EOO}))$
- $\mathsf{EOR}_k = S_B(f_{\mathsf{EOR}_k}, A, B, \ell, k)$
- $\mathsf{Con}_k = S_{TTP}(f_{\mathsf{Con}_k}, A, B, \ell, k)$
- $\mathsf{Abort} = S_A(f_{\mathsf{Abort}}, \ell)$
- $\mathsf{Con}_{abort} = S_{TTP}(f_{\mathsf{Con}_{abort}}, \ell)$

Main Protocol

1. $A \xrightarrow{@} B:\quad f_{\mathsf{Com}}, B, TTP, \ell, c, E_{TTP}(f_{\mathsf{EOO}}, A, B, \ell, k, \mathsf{EOO})$
2. $A \xleftarrow{@} B:\quad f_{\mathsf{EOR}_c}, \ell, \mathsf{EOR}_c$

if A times out *then* abort
else

3. $A \longrightarrow B:\quad f_{\mathsf{EOO}}, A, B, \ell, k, \mathsf{EOO}$

[2] Although the label ℓ contains the identities of both Alice and Bob, we have to add them to the identification, as the TTP is unable to verify the content of the label (the TTP only verifies that the label is coherent with previous messages).

if B times out *then* recovery$[X := B, Y := A]$
else
4. $B \longrightarrow A :$ $\quad f_{\mathsf{EOR}_k}, A, B, \ell, \mathsf{EOR}_k$
if A times out *then* recovery$[X := A, Y := B]$

The main protocol is composed of four messages. Alice starts by sending a commitment to the mail to Bob. This commitment consists of a cipher of the mail m, as well as a cipher c of the key k and the evidence of origin for m, ciphered with the TTP's public key. We also include the identity of the TTP that can be contacted in case of problem, as well as a label ℓ that in conjunction with the identities (A,B) uniquely identifies a protocol run. This label is added to each protocol message and avoids interference of different protocol runs. Moreover the purpose of each message is indicated by a flag. Also note that all the signatures in the protocol are on the hash of the cipher c, and not on the cipher itself. This fact allows us not to send the entire ciphertext, that can be very large, to the TTP in case of a recovery protocol, but only its hash. The first message is sent via an anonymous channel, hiding the sender's identity. Realization of such a channel will be discussed in the next section. Bob replies, via the open anonymous channel, sending an evidence of receipt for the cipher c. If Alice does not receive a valid response before a reasonable amount of time, she launches an abort protocol to stop the protocol. Here, a valid response means a response coherent with the previous message, i.e. the label must match the label in message 1 and the signature has to be correct. Otherwise Alice sends the key k and the evidence of origin EOO to Bob. If Bob does not receive a valid third protocol message, he executes the recovery protocol. Otherwise he sends an evidence of receipt for the key k to Alice. If Alice does not receive a valid evidence of receipt for the key, she has to contact the TTP in order to recover the protocol.

Recovery Protocol

1. $X \longrightarrow TTP :$ $\quad f_{\mathsf{Rec}}, B, \ell, h(c), \mathsf{EOR}_c, E_{TTP}(f_{\mathsf{EOO}}, A, B, \ell, k, \mathsf{EOO})$
if (aborted or recovered) *then* stop
else
recovered=true
2. $TTP \longrightarrow A :$ $\quad f_{\mathsf{Rec}_A}, A, B, \ell, k, \mathsf{Con}_k, \mathsf{EOR}_c$
3. $TTP \longrightarrow B :$ $\quad f_{\mathsf{Rec}_B}, A, B, \ell, k, \mathsf{EOO}$

The aim of the recovery protocol is to enable either Alice or Bob to force a successful end of the protocol. The recovery protocol can be executed, once the protocol has reached a certain state. Bob can recover the protocol once he has got the first message from Alice, and Alice can launch it after having received the second message of the main protocol, i.e. the first message from Bob. When an invalid recovery request arrives, i.e. the signatures do not match the content of the cipher for the TTP, the TTP sends a signed message to alert X that the request is invalid. Receiving an *invalid request* token assures to X that Y will not be able to perform a valid recovery request and X can stop the protocol. Note that an invalid request does not disable the possibility to abort or recover the protocol later. When a valid recovery request arrives, the TTP must first ensure that this protocol run has not been aborted before. The protocol run is identified by the label ℓ and the identities (A,B). The abort protocol and the recovery protocol are mutually exclusive. In a first message either Alice or Bob sends all information the TTP needs to complete the protocol and to be sure the protocol has been started between Alice and Bob. The first message contains the hash of the cipher c, needed for the verification of EOO and EOR_c, the evidence of receipt for the cipher c, as well as the key and the evidence of origin ciphered with the TTP's public key. After having verified the validity of the signatures the TTP sends a confirmation of the key k, replacing the evidence of receipt for k, and the evidence of receipt for c to Alice. The TTP also sends the key k and the evidence of origin to Bob.

Abort Protocol

1. $A \stackrel{@}{\longrightarrow} TTP : E_{TTP}(f_{\mathsf{Abort}}, A, B, \ell, \mathsf{Abort})$
if (recovered or aborted) *then* stop
else
aborted=true
2. $A \stackrel{@}{\longleftarrow} TTP : f_{\mathsf{Con}_{abort}}, \ell, \mathsf{Con}_{abort}$
3. $TTP \longrightarrow B : f_{\mathsf{Con}_{abort}}, \ell, \mathsf{Con}_{abort}$

The abort protocol can be launched by Alice, if Alice does not receive the second message of the main protocol. When Alice aborts the protocol, the TTP first has to verify that the protocol has not yet been recovered. Once the abort protocol has been engaged, recovery is not possible anymore. The communication between Alice and the TTP is ciphered and anonymous, in order to avoid Bob tracing the abort request to its originator. To abort the protocol Alice sends a signed abort request including the label that identifies the protocol. The request is ciphered using the TTP's public key to hide Alice's identity from Bob. The TTP sends a signed abort confirmation to both Alice and Bob. Once Alice started the abort protocol, she must not continue the main protocol anymore, even if the second message arrives. If the second message arrives and Alice continues the protocol, she does not have the ability to recover the protocol anymore. If Bob does not send the last message of the main protocol, Alice will not receive a receipt for her mail. Hence, Alice must stop the protocol after having received an abort token.

Properties

Non-repudiability. The protocol generates a non-repudiation of origin evidence (NRO) and a non-repudiation of receipt evidence (NRR). We have that

– NRO=EOO,

- NRR=$(\mathsf{EOO}_c, \mathsf{EOR}_k)$ or NRR=$(\mathsf{EOO}_c, \mathsf{Con}_k)$.

In case of *repudiation of receipt*, i.e. Bob denies having received a given mail, Alice can prove the receipt by presenting EOR_c, EOR_k or Con_k, EOO, ℓ, m, c, k, as well as the identities of A, B and the TTP to an adjudicator. The adjudicator verifies that:

- $\mathsf{EOO} = S_A(f_{\mathsf{EOO}}, A, B, TTP, \ell, h(c), k)$
- $\mathsf{EOR}_c = S_B(f_{\mathsf{EOR}_c}, B, TTP, \ell, h(c), E_{TTP}(f_{\mathsf{EOO}}, A, B, \ell, k, \mathsf{EOO}))$
- $\mathsf{EOR}_k = S_B(f_{\mathsf{EOR}_k}, A, B, \ell, k)$ or $Con_k = S_{TTP}(f_{\mathsf{Con}_k}, A, B, \ell, k)$
- $\ell = h(m, A, B, k)$
- $c = E_k(m)$

If all the tests hold, the adjudicator concludes that Bob received the message.

In a similar way, in case of *repudiation of origin*, i.e. Alice denies being the author of the message, Bob can present EOO, ℓ, m, c, k, as well as the identities of A, B and the TTP to an adjudicator to prove the mail's origin. The adjudicator verifies that

- $\mathsf{EOO} = S_A(f_{\mathsf{EOO}}, A, B, TTP, \ell, h(c), k)$
- $\ell = h(m, A, B, k)$
- $c = E_k(m)$

If all the tests hold, the adjudicator concludes that Alice is the author of the mail.

Fairness. We will now show that the proposed protocol is fair: neither Alice nor Bob can receive a valuable item without the other one having the possibility to also do so. Therefore we will look at the different possible executions of the protocol. If the main protocol is entirely executed, it is trivial to see that the protocol provides fairness.
We will look at the possible implications of the abort protocol. The abort protocol can only be executed by Alice, as the abort request is signed. Note that a protocol is identified by ℓ and $(A, B)(\neq (B, A))$. Hence, this protocol run can only be aborted by Alice. Alice has the possibility to execute the abort protocol at any moment. However, executing it after having sent the third message of the main protocol could harm Alice. We therefore suppose that a honest Alice only executes the abort protocol before the third message of the main protocol. That means that neither the mail m nor any of the non-repudiation evidences has been exchanged. Any recovery request from either Alice or Bob will be refused by the TTP. In order not to harm herself, Alice will not continue the main protocol, once she executed the abort protocol. There is no possibility, neither for Alice nor Bob, to receive any valid item in this protocol run.
We will continue examining the consequences of a recovery protocol. Mutual exclusion of the abort protocol and the recovery protocol is ensured by the TTP.

We can not have a situation where Alice has stopped the protocol, after having aborted it and Bob can recover the protocol by receiving the mail and the non-repudiation of origin message. The recovery protocol sends the expected items to both Alice and Bob. The only way of breaking fairness would be to send an invalid item to one of the entities and not to the other. However sending any invalid item will result in invalid evidences. Consider for instance, that Alice sends an invalid key in the first message of the main protocol. This key is always included in the non-repudiation of receipt evidence and so the evidence will be invalid.
Now consider the case where an invalid recovery request is sent to the TTP. Bob cannot verify the validity of his recovery request and could possibly not be able to generate a valid recovery request, as a part of it has been ciphered by Alice for the TTP. If the cipher contains incorrect information, the recovery cannot be performed and Bob is informed of this by the mean of an incorrect request token. However in that case neither Alice nor Bob can perform a recovery, as the cipher containing incorrect data has been signed by Bob in EOR_c and can not be replaced by Alice. We conclude that our protocol provides fairness.

Timeliness. Timeliness is provided by the fact that the communication channels between the TTP and both Alice and Bob are resilient. This means that all sent messages are received correctly after a finite amount of time. Looking at the protocol, we see that Alice at each moment of the protocol can contact the TTP to end the protocol: before having sent the third message Alice can abort the protocol and thereafter Alice can recover it. Bob can always execute a recovery protocol or at least receive an incorrect request token signed by the TTP. Hence at any moment in the protocol, both Alice and Bob have the ability to finish the protocol in a finite amount of time. In the previous paragraph we showed that executing the abort or the recovery protocol results in a fair termination of the protocol. Hence our protocol provides timeliness.

No selective receipt. No message based selective receipt is implied by fairness. To show that the protocol respects *no author based selective receipt*, we look at all possible executions of the protocol. After arrival of the first message, as the transmission is anonymized, Bob does not know Alice's identity. At this moment, Bob has the possibility to either execute the recovery protocol, stop the protocol or continue it by sending the second message. If Bob launches the recovery protocol, the protocol ends succesfully, so Alice's identity may be revealed. If Bob stops the protocol after having received the first message, Alice will execute the abort protocol. The abort protocol needs to be anonymized to avoid the following attack: Bob stops the protocol after having received the first message and waits for the TTP to receive an abort request for the protocol. Even if this request is ciphered, one could try to trace all incoming requests and recover Alice's identity. Therefore we use anonymous transmissions that can not be traced. All data sent to an anonymity provider must also be ciphered for this provider. For instance the abort request, is first ciphered for the TTP, and then, additionally, for the anonymity provider. Otherwise, Bob can permanently observe several “potential Alices”, and compare outgoing messages to the certified e-mails he is receiving and to the requests arriving at the TTP. If Bob sends the second message, Alice will send message 3 of the main protocol. Once the third message is sent the fairness property ensures succesful termination of the protocol.

3.2 A confidential certified e-mail protocol

Notations The notation will be the same as in the previously described protocol. The changed evidences generated in the protocol are the following.

- $\mathsf{EOO} = S_A(f_{\mathsf{EOO}}, A, B, TTP, \ell, h(c), E_B(k))$
- $\mathsf{EOR}_c = S_B(f_{\mathsf{EOR}_c}, B, TTP, \ell, h(c), E_{TTP}(f_{\mathsf{EOO}}, A, B, \ell, k, \mathsf{EOO}))$
- $\mathsf{EOR}_k = S_B(f_{\mathsf{EOR}_k}, A, B, \ell, E_B(k))$
- $\mathsf{Con}_k = S_{TTP}(f_{\mathsf{Con}_k}, A, B, \ell, E_B(k))$

The main protocol is very similar to the previous protocol. The difference with the first protocol is that the key k is systematically ciphered using Bob’s public key. There are two reasons to do so. Firstly, we avoid an additional ciphering operation at the TTP during a recovery request. Secondly, we avoid that the TTP could gain knowledge or help some external attacker to gain knowledge of the mail. The only change we made to the recovery protocol, is that the first message contains the key ciphered with Bob’s public key, and in messages 2 and 3 the ciphered key is sent instead of the plain key. Neither the TTP, nor an external observer, ever gains knowledge of the mail content. The abort protocol is identical to the abort protocol in the previous section.

Properties Most of the reasonings for the previously discussed properties also hold for this protocol, as only minimal changes have been done. When we contact the adjudicator, we have to send all data on a confidential channel, in order for the mail content to stay confidential. When the adjudicator verifies the evidences, he additionally needs to check that ciphering k with Bob’s public key results to $E_k(B)$ signed in the respective evidences. Therefore we must not use a probabilistic ciphering algorithm. We need to use a ciphering algorithm, that associates exactly one cipher to one plain text. Confidentiality is easily proved. The plain mail is never sent over the network. To recover the mail, one has to know the cipher c and the key k. However, the key k only intervenes ciphered under Bob’s public key. If the ciphering algorithms producing both c and $E_B(k)$ are secure, the only party that can gain knowledge of m is Bob.

4 Anonymous Transmissions

In this section we will briefly discuss how anonymous transmissions may be realized. Consider first a situation where the network address does not reveal someone's identity. We can be the case of public places offering an Internet access, e.g. a cybercafe. In that case no additional mechanisms to provide anonymity are required.

In many cases, the network address can be thought to be equivalent to a person's identity. Then we need more elaborate mechanisms. A simple and commonly used way to provide anonymity on a network consists in special hosts, such as *anonymizers* or *re-mailers*. These hosts are used as an intermediate to directly hide the link between two hosts. Although this solution is very easily implemented, it suffers from several drawbacks. It does not resist more elaborate attacks, based on traffic analysis. Examples of such attacks are described in [5].

If we have to care about a powerful receiver, that can attack simple anonymizers, one has to use solutions, such as *mixnets*, proposed by Chaum in [6]. Mixnets are chains of anonymizers, called *mixes* that only know the address of the preceding and the following host. All sent message blocks have same length and the hosts always wait for a lower bound of messages before forwarding them. The order of the different incoming messages is mixed, making tracing of messages impossible. Moreover, $k-1$ out of k mixes may collude, without compromizing anonymity. Although mixnets are rather inefficient, they offer secure anonymity services.

However, in our context, mixnets are not suitable, as they do not offer the possibility of *anonymous replies*: in our protocols, Bob has to reply to Alice on an anonymous channel without knowing her identity. On simple re-mailers, solutions based on pseudonyms are available. If we need secure anonymity services, resisting traffic analysis we may use the *Onion Routing* system [12], proposed and maintained by the US Naval Research Center. The onion routing system is based on an anonymous connection, set up once for the whole transmission, and defines a solution to the anonymous reply problem.

In practice a trade-off between security and efficiency needs to be made. The choice of the chosen solution will depend on the importance of the mail.

5 Conclusion

In this paper we presented a new protocol for certified e-mail. We discussed the properties of certified e-mail and their links to related problems, namely fair exchange, contract signing and non-repudiation. We introduced a new property, no author based selective receipt, that is specific to certified e-mail. The property claims that once the identity of the author of a mail is known, the receipt of the mail can not be refused anymore. Then we presented two variants of a new protocol: the first one does not provide confidentiality, while the second one does. As confidentiality is not always required, and harms the efficiency of the protocol, we suggest to leave the choice to the user, whether confidentiality is provided or not. Both protocols provide no author-based selective receipt. Finally, we discuss some mechanisms to provide anonymous transmissions that are needed in our protocols.

References

1. N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In T. Matsumoto, editor, *4th ACM Conference on Computer and Communications Security*, pages 8–17, Zurich, Switzerland, Apr. 1997. ACM Press.
2. N. Asokan, V. Shoup, and M. Waidner. Asynchronous protocols for optimistic fair exchange. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 86–99, Oakland, CA, May 1998. IEEE Computer Society, Technical Committee on Security and Privacy, IEEE Computer Society Press.
3. G. Ateniese, B. de Medeiros, and M. T. Goodrich. TRICERT: A distributed certified E-mail scheme. In *Symposium on Network and Distributed Systems Security (NDSS 2001)*, San Diego, CA, Feb. 2001. Internet Society.
4. M. Ben-Or, O. Goldreich, S. Micali, and R. L. Rivest. A fair protocol for signing contracts. *IEEE Transactions on Information Theory*, 36(1):40–46, 1990.
5. O. Berthold, H. Federrath, and M. Köhntopp. Anonymity and unobservability in the internet. In *Workshop on Freedom and Privacy by Design*, Toronto, Canada, Apr. 2000.
6. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, Feb. 1981.
7. R. H. Deng, L. Gong, A. A. Lazar, and W. Wang. Practical protocols for certified electronic mail. *Journal of Network and System Management*, 4(3), 1996.
8. S. Kremer and O. Markowitch. Optimistic non-repudiable information exchange. In J. Biemond, editor, *21st Symp. on Information Theory in the Benelux*, pages 139–146, Wassenaar (NL), May25-26 2000. Werkgemeenschap Informatie- en Communicatietheorie, Enschede (NL).
9. O. Markowitch and Y. Roggeman. Probabilistic non-repudiation without trusted third party. In *Second Conference on Security in Communication Networks'99*, Amalfi, Italy, Sept. 1999.
10. J. Riordan and B. Schneier. A certified E-mail protocol. In *14th Annual Computer Security Applications Conference*. ACM, 1998.
11. M. Schunter. *Optimistic Fair Exchange*. PhD thesis, Technische Fakultät der Universität des Saarlandes, Saarbrücken, 2000.
12. P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. In *IEEE Symposium on Security and Privacy, May 4–7, 1997, Oakland, California*, pages 44–54. IEEE Computer Society Press, 1997.
13. T. Tedrick. Fair exchange of secrets. In G. R. Blakley and D. C. Chaum, editors, *Advances in Cryptology: Proceedings of Crypto'84*, volume 196 of *Lecture Notes in Computer Science*, pages 434–438. Springer-Verlag, 1985.
14. J. Zhou, R. Deng, and F. Bao. Evolution of fair non-repudiation with TTP. In *ACISP: Information Security and Privacy: Australasian Conference*, volume 1587 of *Lecture Notes in Computer Science*, pages 258–269, 1999.
15. J. Zhou and D. Gollmann. Certified electronic mail. In *Proc. ESORICS '96*, volume 1146 of *Lecture Notes in Computer Science*, pages 160–171, 1996.
16. J. Zhou and D. Gollmann. An efficient non-repudiation protocol. In *PCSFW: Proceedings of The 10th Computer Security Foundations Workshop*. IEEE Computer Society Press, 1997.