A Formal Analysis of Fairness and Non-repudiation in the RSA-CEGD Protocol

Almudena Alcaide, Juan M. Estévez-Tapiador,
Antonio Izquierdo, and José M. Sierra

Department of Computer Science, Carlos III University of Madrid,
Avda. Universidad 30, 28911, Leganés, Madrid (Spain)
{aalcaide, jestevez, aizquier, sierra}@inf.uc3m.es

Abstract. Recently, Nenadić *et al.* (2004) proposed the RSA-CEGD protocol for certified delivery of e-goods. This is a relatively complex scheme based on verifiable and recoverable encrypted signatures (VRES) to guarantee properties such as strong fairness and non-repudiation, among others. In this paper, we illustrate how an extended logic of beliefs can be helpful to analyze in a formal manner these security properties. This approach requires the previous definition of some novel constructions to deal with evidences exchanged by parties during the protocol execution. The study performed within this framework reveals the lack of non-repudiation in RSA-CEGD and points out some other weaknesses.

1 Introduction

Interest in protocols for fair exchange of information with non-repudiation stems from its importance in many applications where disputes among parties can occur. Assurance of these properties enables the deployment of a wide range of applications, such as certified e-mail or business transactions through communication networks. As a result, fair non-repudiation has experienced an explosion of proposals in recent years (see [9] for an excellent survey).

Nevertheless, fairness and non-repudiation have not been so extensively studied as other classic issues, such as confidentiality or authentication. Previous experience in these contexts has shown that designing security protocols is an error-prone task, and that formal analysis can aid in detecting weaknesses. However, many of the protocols proposed to achieve fair non-repudiation have not been subject to a formal security analysis. In the field of security protocols, formal methods have mainly focused on the analysis of authentication and key-establishment protocols. These techniques, however, cannot be directly applied to reasoning about properties such as fairness or non-repudiation without a previous formalization of what the protocol goals are. Furthermore, many assumptions usually done for other protocols are no longer valid in these environments. For instance, most of the analysis models developed consider the existence of an attacker with different capabilities, e.g. control over the communication channels, though the protocol parties are assumed to trust each other and behave

O. Gervasi et al. (Eds.): ICCSA 2005, LNCS 3483, pp. 1309–1318, 2005.

according to the protocol rules. Nevertheless, in fair exchange or non-repudiation there is no adversary per se [15], such as the classic Dolev-Yao intruder model [4]. These protocols, on the contrary, are designed for scenarios wherein participants may misbehave with the aim of cheating the other party, thus obtaining her own profit.

The tools developed to verify fair non-repudiation are scarce when compared to those available for classic security requirements. To the best of our knowledge, the first attempt to reasoning about this kind of protocols in a formal manner was due to Kailar [6, 7]. He used a BAN-like logic wherein the central construction is the predicate `CanProve`, as in "A `CanProve` B `says` X". Properties such as non-repudiation can be easily formalized using this predicate. A similar approach was introduced by Zhou and Gollman in [17]. In this work, however, authors opted for the use of the SVO logic instead of BAN. An alternative was presented by Schneider in [14], in which a verification of a non-repudiation protocol is carried out by using CSP (*Communicating Sequential Processes*). This is an abstract language aimed at modeling the communication patterns of concurrent systems that interact through message passing.

Even though evidences of non-repudiation can be provided by means of classic cryptographic techniques, fairness is much more difficult to achieve and also to guarantee. Recent works due to Kremer and Raskin have focused on the consideration of exchange protocols as a game wherein messages exchanged by participants can be seen as their strategies [10, 11].

Despite efforts as those mentioned above, it still remains difficult to assure formally that a protocol guarantees fair non-repudiation. Consider, as an illustrative example, a protocol proposed in 1996 by Zhou and Gollman [16] that was verified and proved correct using three different methods [2, 14, 17]. Surprisingly, in 2002 Gürgens and Rudolph demonstrated the absence of fair non-repudiation in that protocol under reasonable assumptions [5]. In this case, possible attacks were detected after an analysis performed with a different formalism that considered scenarios not checked before.

Verifications of security properties must be carefully treated. A formal analysis does not establish security, but only with respect to a model which, in turn, is based on assumptions. Many of these assumptions may be violated in scenarios not considered by the protocol designers. As a result, assumptions are usually the inherent weaknesses, exactly in the sense pointed out by Denning: "the way to crack a system is to step outside the box, to break the rules under which the protocol was proved secure" [3]. Anyway, the formalization effort required to perform a verification serves to highlight assumptions and can clarify some flaws in the protocol design, thus pointing out possible vias of exploitation. In this paper, we show how an extended logic of beliefs can be useful to reasoning about properties such as fairness and non-repudiation. Our aim with this approach is not to prove security, but to: 1) identify critical steps in the protocol evolution; 2) analyze the evidences supporting security properties that each party possesses; and 3) reasoning about scenarios in which such evidences might be misused with success. As a proof-of-concept, we illustrate this methodology

by presenting a partial analysis of the RSA-CEGD protocol [12] –an RSA-based scheme for certified e-goods delivery. Due to space restrictions, in this work we focus exclusively in the strong fairness and non-repudiation properties, showing that the latter is not satisfied.

The rest of this paper is organized as follows. For completeness, Section 2 presents a brief overview of the RSA-CEGD protocol. Section 3 introduces a formalization of the two features of this protocol considered in this work: strong fairness and non-repudiation. The core of the analysis is exposed in Section 4, while Section 5 discusses in detail the key points of our approach. Finally, Section 6 concludes the paper.

2 Overview of the RSA-CEGD Protocol

Throughout this paper, we will use the same notation introduced by the authors in the original paper [12]. The RSA-CEGD is an optimistic fair exchange protocol composed of two sub-protocols, as shown in Fig. 1. As usual, the exchange sub-protocol is used to carry out the exchange between parties without any TTP's involvement. In case the process fails to complete successfully, a recovery protocol can be invoked to handle this situation.

The notion of verifiable and recoverable encrypted signature (VRES) underlies at the core of the RSA-CEGD protocol. A VRES is basically an encrypted signature, which acts as a *receipt* from the receiver's point of view, with two main properties. First, it can be *verified*: the receiver is assured that the VRES contains the expected signature without obtaining any valuable information about the signature itself during the verification process. And second, the receiver is assured that the original signature can be *recovered* with the assistance of a designated TTP in case the original sender refuses to do it.

Due to these two properties, the VRES becomes an interesting cryptographic primitive upon which fairness can be provided. The RSA-CEGD protocol relies on this element within the general scheme we sketch in what follows:

1. A ciphers the message with an encryption key and sends it to B.
2. B generates the VRES of his signature and sends it back to A.
3. Upon successful verification of the VRES, A is assured that it is secure for her to send the decryption key to B, so he can access the message.
4. Finally, B sends his original signature to A as a receipt. In case he refuses, a TTP can recover the signature from the VRES, thus restoring fairness.

The RSA-CEGD protocol makes use of a novel VRES method based on the RSA system, hence its name. The idea stems from the so-called *theory of cross-decryption* [13], which establishes that an RSA encrypted text can be decrypted by using two different keys if both pairs of secret/public keys are appropriately chosen. Party B is enforced to use a key of this kind to encrypt the VRES, while the TTP retains the other. This way, if subsequently B refuses to provide A with his signature, the TTP is able to recover it from the VRES.

The exchange sub-protocol
E1: $\mathtt{P_a} \rightarrow \mathtt{P_b} : E_{k_a}(D_a),\ CertD_a,\ x_a,\ E_{sk_a}(h_a)$
E2: $\mathtt{P_b} \rightarrow \mathtt{P_a} : (x_b, xx_b, y_b),\ s_b,\ C_{bt}$
E3: $\mathtt{P_a} \rightarrow \mathtt{P_b} : r_a$
E4: $\mathtt{P_b} \rightarrow \mathtt{P_a} : r_b$

The recovery sub-protocol
R1: $\mathtt{P_a} \rightarrow \mathtt{P_t} : C_{bt},\ y_b,\ s_b,\ y_a,\ r_a$
R2: $\mathtt{P_t} \rightarrow \mathtt{P_a} : r_b$
R3: $\mathtt{P_t} \rightarrow \mathtt{P_b} : r_a$

Definition of the protocol's items

$x_a = (r_a \times k_a)\ mod\ n_a$: encryption of P_a's key k_a with random number r_a

$CertD_a = (desc_a, hd_a, h_a, ek_a, sign_{CA})$: certificate for D_a issued by the CA, where
$desc_a$ = description (content summary) of D_a
$hd_a = h(E_{k_a}(D_a))$: hash value of the encryption of D_a with the key k_a
$h_a = h(D_a)$: hash value of D_a
$ek_a = E_{pk_a}(k_a)$: encryption of the key k_a with P_a's public key, pk_a

$E_{sk_a}(hd_a)$: P_a's RSA signature on D_a serving as a proof of origin of D_a

r_a : random prime generated by P_a 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 : random prime generated by P_b for the generation of the VRES (y_b, x_b, xx_b)

$rec_b = (h_a)^{d_b}\ mod\ n_b$: P_b's receipt for P_a's e-goods D_a, i.e. P_b's RSA signature on D_a

(y_b, x_b, xx_b): P_b's VRES, where
$y_b = r_b{}^{e_b}\ mod\ (n_b \times n_{bt})$: encryption of r_b with P_b's public key. Also recoverable by P_t
$x_b = (r_b \times (h_a)^{d_b})\ mod\ n_b = (r_b \times rec_b)\ mod\ n_b$: encryption of rec_b with r_b
$xx_b = (r_b \times E_{sk_{bt}}(h(y_b)))\ mod\ n_{bt}$: control number that confirms the correct use of r_b

$C_{bt} = (pk_{bt}, w_{bt}, s_{bt})$: P_b's RSA public-key certificate issued by P_t
$pk_{bt} = (e_{bt}, n_{bt})$: public RSA key related to C_{bt}, with $e_{bt} = e_b$
$sk_{bt} = (d_{bt}, n_{bt})$: private RSA key related to C_{bt}
$w_{bt} = (h(sk_t, pk_{bt})^{-1} \times d_{bt})\ mod\ n_{bt}$
$s_{bt} = E_{sk_t}(h(pk_{bt}, w_{bt}))$: P_t's signature on $h(pk_{bt}, w_{bt})$.

$s_b = E_{sk_b}(h(C_{bt}, y_b, y_a, P_a))$: P_b's recovery authorization token

Fig. 1. The RSA-CEGD protocol

3 Formalization of Protocol Objectives

The protocol RSA-CEGD presented in [12] is defined to satisfy a list of six different objectives. In this paper, we will only focus on the formal analysis of three of them, the ones related to strong fairness and non-repudiation. The specific definitions for each of those three goals are quoted literally from [12]. Moreover, our analysis focuses on the items considered by the protocol authors as proof of fairness and non-repudiation. In the following section, such objectives and evidences will be formalized using classic BAN notation.

3.1 Strong Fairness and Correctness

Strong fairness is defined by the protocol authors as follows: *"If and only if 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 them with the assistance of the STTP"*. Consequently, the protocol is fair and correct if and only if the following formulae are verified:

- If P_b has got the goods, then P_a has the authorization token and the item $\langle rec_b \rangle_{r_b}$:

$$\frac{P_b \triangleleft D_a}{P_a \models s_b, \quad P_a \triangleleft \langle rec_b \rangle_{r_b}} \tag{1}$$

- If P_a has rec_b, then P_b has the goods:

$$\frac{P_a \triangleleft rec_b}{P_b \triangleleft D_a} \tag{2}$$

3.2 Non-repudiation of Origin of the E-goods (NRO)

As stated when defining the objectives for the RSA-CEGD, non-repudiation of origin is preserved when: *"The recipient is provided with a proof that the sender is indeed the originator of the e-goods"*. The protocol establishes the item $\{hash(D_a)\}_{sk_a}$ as the only proof of origin. To preserve NRO, P_b must be sure that P_a is the real sender of such a message, and P_a must also ensure that P_b cannot have access to such a message if P_a has not sent it. This is, NRO is preserved when these two formulae are satisfied:

$$\frac{P_b \triangleleft \{hash(D_a)\}_{sk_a}}{P_b \models P_a \models \{hash(D_a)\}_{sk_a}} \tag{3}$$

$$\frac{P_b \triangleleft \{hash(D_a)\}_{sk_a}}{P_a \models \{hash(D_a)\}_{sk_a}} \tag{4}$$

Regarding formula (3), P_b knows that the originator of the e-goods is P_a. However, P_b knows that the sender is indeed P_a, *only* because the channel between P_a and P_b is considered to be authenticated and confidential. Therefore, prior to the current run of the protocol P_a and P_b have been authenticated as the parties involved in the communication, and P_b can be sure that the sender of the signature that he receives is P_a. The formula $\{hash(D_a)\}_{sk_a}$ itself does not prove that P_a is the sender as well as the originator. Furthermore –formula (4)–, P_a cannot know whether P_b can obtain the proof of origin from a different source. The formula $\{hash(D_a)\}_{sk_a}$ may not belong to the current execution. P_b could make P_a responsible for a item that P_a never sent to him. The validation process will show how NRO cannot be verified.

3.3 Non-repudiation of Receipt of the E-goods (NRR)

RSA-CEGD authors consider the property of non-repudiation of receipt to be preserved when *"The sender of the e-goods is provided with a proof that the intended recipient has indeed received the e-goods"*. In this case, P_b's signature over the hash value of the e-goods is admitted as irrevocable proof that P_b received the e-goods. Therefore, NRR is preserved if the following formula is satisfied:

$$\frac{P_a \triangleleft \{hash(D_a)\}_{sk_b}}{P_a \models P_b \triangleleft D_a} \tag{5}$$

Here, the item $\{hash(D_a)\}_{sk_b}$ does link the entity P_b and the e-goods D_a. On the other hand, $CertD_a$ will join together the e-goods D_a and the originator entity P_a.

4 Security Analysis

The entire validation process is described in detail in [1]. In what follows, only the necessary steps for analyzing the objectives are shown.

4.1 Analysis of Strong Fairness and Correctness

As stated in Section 3.1, the protocol is fair and correct if and only if formulae (1) and (2) are verified. In the former case, the most important steps of its analysis are provided below:

1. Message	E1: $\mathtt{P_a} \rightarrow \mathtt{P_b}$:	$E_{k_a}(D_a),\ CertD_a,\ x_a,\ E_{sk_a}(h_a)$
2. Items		$\{\mathbf{D_a}\}_{\mathbf{K_a}}, \mathbf{CertD_a}, \langle \mathbf{K_a} \rangle_{\mathbf{r_a}}, \{\mathbf{h_a}\}_{\mathbf{sk_a}}$
3. Message	E2: $\mathtt{P_b} \rightarrow \mathtt{P_a}$:	$(x_b, xx_b, y_b),\ s_b,\ C_{bt}$
4. Conclusions		$\mathbf{P_a} \triangleleft \langle \mathbf{rec_b} \rangle_{\mathbf{r_b}}$, where $rec_b = \{h_a\}_{sk_b}$
		$\mathbf{P_a} \models \mathbf{s_b}$, where $s_b = \{hash(C_{bt}, y_b, y_a, P_a)\}_{sk_b}$
5. Message	E3: $\mathtt{P_a} \rightarrow \mathtt{P_b}$:	r_a
6. Conclusions		$\mathbf{P_b} \triangleleft \mathbf{D_a}$

The sequential order in which the exchange protocol is executed gives us the implication: $P_b \triangleleft D_a \Rightarrow P_a \models s_b, P_a \triangleleft \{rec_b\}_{r_b}$. Otherwise, P_t's honesty and *resilient communication channels* between P_b and P_t would ensure the property when the recovery protocol is invoked.

The analysis concerning formula (2) is completely analogous. In a similar way, the sequential order in which the exchange protocol is executed gives us the implication: $P_a \triangleleft rec_b \Rightarrow P_b \triangleleft D_a$. Again, P_t's honesty and *resilient communication channels* between P_a and P_t would ensure the property in case the recovery protocol has to be invoked.

Finally, note that due to the absence of any timeout routine as part of the protocol, a deadlock conflict could arise easily. For example, a network failure can prevent P_a from receiving message E2, while at the same time P_b –unaware of the failure– keeps awaiting to receive message E3.

4.2 Analysis of NRO

According to the definition provided in Section 3.2, NRO is verified if formulae (3) and (4) hold. In the former case, the most important steps of its analysis are provided in what follows:

1. Message	E1: $\mathtt{P_a} \rightarrow \mathtt{P_b}$:	$E_{k_a}(D_a),\ CertD_a,\ x_a,\ E_{sk_a}(h_a)$
2. Items		$\mathbf{P_b} \models \mathbf{P_a} \mid\sim \mathbf{h_a}$
3. Conclusions		$\mathbf{P_b} \triangleleft \mathbf{h_a}$, as h_a is part of $CertD_a$
4. Items		$\mathbf{P_b} \models \overset{+\mathbf{pk_a}}{\mapsto} \mathbf{P_a} \Leftrightarrow \mathbf{P_b} \triangleleft \mathbf{h_a}$
5. Conclusions		P_b verifies $h_a = hash(D_a)$ from $CertD_a$ Therefore: $\mathbf{P_b} \triangleleft \{\mathbf{h_a}\}_{\mathbf{sk_a}}$
6. Conclusions		So P_b knows that P_a is the originator: $\mathbf{P_b} \models \mathbf{P_a} \mid\sim \{\mathbf{h_a}\}_{\mathbf{sk_a}}$
7. Conclusions		Because the channel between P_a and P_b is considered to be authenticated and confidential, P_b knows that P_a is the sender. So P_a must *still believe* in such a message: $\mathbf{P_b} \models \mathbf{P_a} \models \{\mathbf{hash(D_a)}\}_{\mathbf{sk_a}}$

Therefore, the formula $\{hash(D_a)\}_{sk_a}$ itself does not prove that P_a is the sender as well as the originator. The item $\{hash(D_a)\}_{sk_a}$ could have been used before, i.e. in another instance of the same protocol. An entity, different from P_a, could have been part of a genuine run of the protocol and could now be reusing such a formula with malicious purposes (replay attack).

Furthermore, neither the implication (4) can be verified with the items provided by each party during the protocol execution, nor confidentiality of communications can assists in the validation process of implication (4). P_a cannot know whether P_b can obtain a proof of origin from a different source. In fact, the formula $\{hash(D_a)\}_{sk_a}$ may not belong to the current protocol execution. P_b could be reusing the item $\{hash(D_a)\}_{sk_a}$ obtained from an entity which participated with P_a in a previous run of the protocol. In such a case, P_b was not meant to be the receiver of item $\{hash(D_a)\}_{sk_a}$, even though it is possible that P_b could make P_a responsible for having sent a proof of origin that P_a never sent to P_b. As a result, NRO is not preserved.

4.3 Analysis of NRR

According to the definition given in Section 3.3, satisfying the double implication expressed by formula (5) will verify NRR. In this case, the sequential order in which the exchange protocol is executed gives us the implication: $P_a \triangleleft \{h_a\}_{sk_b} \Rightarrow P_b \triangleleft D_a$. Otherwise, *resilient communication channels* between P_b and P_t would ensure the property when the recovery protocol is invoked.

Finally, note that the communications channels between P_t and entities P_a and P_b, coupled with P_t's honesty, play a crucial role with regard to NRR and fairness.

5 Discussion

5.1 On the Analysis Process

The strong fairness and non-repudiation protocol RSA-CEGD has been formally analyzed using a BAN-like logic. The entire validation and analysis processes are described in [1]. BAN and related logics were never designed to validate security properties such as fairness or non-repudiation, but authentication and key-exchange protocols. However, the modifications and new rules added to BAN to pursue our goals have been minimal. Only new concepts as public keys certificates or VRES signatures have been described using new rules in order to formalize their definitions and behaviors.

A new operator has also been introduced to define the double implication "if and only if" between formulae. The double implication has been represented as a double line between formulae within the same rule.

5.2 Double Implication in Logic Rules

Double implication in the extended rules is introduced to gain completeness as well as correctness of the validation process. Fig. 2(a) depicts a general stage of the protocol validation process. For each stage, a set of hypotheses H_A is defined for a given entity A. Within that set H_A, A carries out some verifications and computations. If the verifications are negative then A will abort the current run of the protocol. Otherwise, if the verifications are positive, A can use the extended set of rules to infer the set $Results_A$ with a double implication, i.e: "if and only if".

The *correctness* of such a stage will be verified if A can obtain the expected set of objectives from the set H_A and the verifications carried out in it; but also, if A can prove that no other type of verifications or computations could result in contradicting the objectives. In this regard, the process ensures A that if results contradicting the set $Results_A$ can be derived from a different set of computations, then $Verifications_A$ would not have been satisfied, so A would have aborted the execution of the protocol. This situation is graphically explained in Fig. 2(c). Note that the set of hypotheses H_A is considered to be well defined and determined by the protocol. Only secret keys, public keys and certificates are part of the initial set H_A. For each stage, the formulae contained within the message sent and the set of results from previous stages are added to H_A.

The *completeness* of the validation process comes from the double implication. How can A be sure that, given the hypotheses in set H_A, a different set of verifications and computations, not related to the ones determined by the protocol, could not reach to the same results as in set $Results_A$?. This could be the case when an entity is trying to cheat on another or when an attacker is trying to carry out an attack. The situation is shown in the diagram depicted in Fig. 2(b). In this situation, the double implication would mean that $Verifications'_A \Rightarrow Verifications_A$, or equivalently $Verifications'_A \subseteq Verifications_A$. This way, any different set of verifications and computations should be part –or imply– those determined by the protocol.

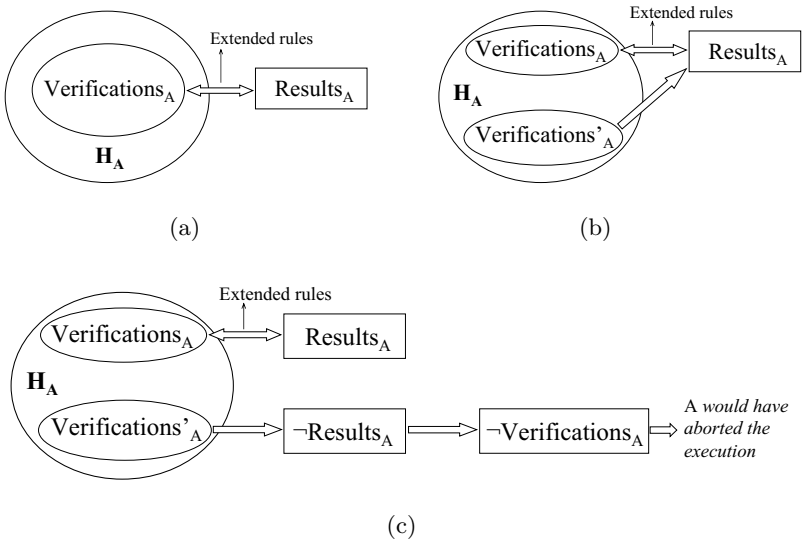

Fig. 2. Scheme of the analysis process

5.3 Reasoning About Evidences

Suppose entities P_a and P_b, and a message X being sent from P_a to P_b. The concept of freshness, represented in BAN as $\sharp(X)$, is the only operator to promote a belief such $P_a \mid\sim X$ onto $P_a \mid\equiv X$. This is, "P_a once said X" onto "P_a still believes on X". We have seen in Section 4.2 how the formula $P_a \mid\sim X$ is not enough to prove origin of message X. Any given proof of origin should somehow link the sender, the originator of X and X itself with the undergoing protocol execution.

Something very similar applies to non-repudiation of receipt. A proof of receipt should link together the receiver and the message received with the undergoing protocol execution. In fact, these requirements are well understood in the design of non-repudiation protocols (useful guidelines can be found in [8, 9]).

6 Conclusions

In the approach outlined in this paper, we have attempted to reasoning formally about some security properties of a complex protocol, such as it is the case of RSA-CEGD. Formalization provides us with an useful framework for identifying weaknesses that can get easily unnoticed during an informal study. Moreover, the very process of analysis serves frequently not only to point out possible attacks, but also aids to understand how the protocol can be improved.

References

1. A. Alcaide and J. M. Estévez. "Formal Analysis of the RSA-CEGD protocol". *Technical Report*, January 2005.
2. G. Bella and L. Paulson. "Mechanical Proofs about a Non-repudiation Protocol". *Proc. 14th Intl. Conf. Theorem Proving in Higher Order Logic*. LNCS, pp.91–104. Springer-Verlag, 2001.
3. D. E. Denning. "The Limits of Formal Security Models". National Computer Systems Security Award Aceptance Speech. October, 1999. Available online at: `http://www.cs.georgetown.edu/~denning/infosec/award.html`.
4. D. Dolev and A. C. Yao. "On the Security of Public Key Protocols", *IEEE Trans. Inf. Theory*, IT-29(12):198–208, 1983.
5. S. Gürgens and C. Rudolph. "Security Analysis of (Un-) Fair Non-repudiation Protocols", *FASec 2002*, LNCS 2629, pp. 97–114. Springer-Verlag, 2002.
6. R. Kailar. "Reasoning about accountability in protocols for electronic commerce". *Proc. IEEE Symp. Security and Privacy*, pp. 236–250. IEEE Computer Security Press, 1995.
7. R. Kailar. "Accountability in electronic commerce protocols". *IEEE Trans. Software Engineering*, 5(22):313–328, 1996.
8. P. Louridas. "Some guidelines for non-repudiation protocols". *Computer Communication Review*, 30(5):29–38. October, 2000. ACM Press.
9. S. Kremer, O. Markowitch, and J. Zhou. "An intensive survey of fair non-repudiation protocols". *Computer Comunications*, 25(17):1606–1621. Elsevier Science B.V., 2002.
10. S. Kremer and J. F. Raskin. "A game approach to the verification of exchange protocols - application to non-repudiation protocols". *Workshop on Issues in the Theory of Security (WITS'00)*, July 2000.
11. S. Kremer and J. F. Raskin. "A Game-Based Verification of Non-Repudiation and Fair Exchange Protocols". *Journal of Computer Security*, 11(13):399–429, 2003.
12. A. Nenadić, N. Zhang, S. Barton. "A Security Protocol for Certified E-goods Delivery", *Proc. IEEE Int. Conf. Information Technology, Coding, and Computing (ITCC'04)*, Las Vegas, NV, USA, IEEE Computer Society, 2004, pp. 22–28.
13. I. Ray and I. Ray. "An Optimistic Fair Exchange E-commerce Protocol with Automated Dispute Resolution". *Proc. Int. Conf. E-Commerce and Web Technologies, EC-Web 2000*. LNCS 1875, pp. 84–93. Springer-Verlag, 2000.
14. S. Schneider. "Formal Analysis of a Non-repudiation Protocol". *IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, 1998.
15. P. Syverson and I. Cervestato. "The Logic of Authentication Protocols", R. Forcardi and R. Gorrieri, Eds.: *Foundations of Security Analysis and Design (FOSAD'00)*, Tutorial Lectures, LNCS 2171, pp. 63–136. Springer-Verlag, 2000.
16. J. Zhou and D. Gollman. "A fair non-repudiation protocol". *Proc. 1996 Symp. on Research in Security and Privacy*, pp. 55–61. Oakland, CA, USA. IEEE Computer Society Press, 1996.
17. J. Zhou and D. Gollman. "Towards verification of non-repudiation protocols". *Proc. 1998 Intl. Refinement Workshop and Formal Methods Pacific*, pp. 370–380. 1998.