Chapter 3

A FAIR-EXCHANGE E-COMMERCE PROTOCOL WITH AUTOMATED DISPUTE RESOLUTION

Indrajit Ray
Department of Computer and Information Science
University of Michigan-Dearborn
indrajit@umich.edu

Indrakshi Ray
Department of Computer and Information Science
University of Michigan-Dearborn
iray@umich.edu

Natarajan Narasimhamurthi
Department of Electrical and Computer Engineering
University of Michigan-Dearborn
nnarasim@engin.umd.umich.edu

Abstract

In this paper, we present a fair-exchange electronic commerce (e-commerce) protocol, based on using an online trusted third party, that ensures fairness and prevents any party from gaining advantage by quitting prematurely from the transaction or otherwise misbehaving. An important contribution of this protocol is that the dispute resolution is taken care of within the protocol itself and does not require manual intervention. Thus even if one of the parties disappear after the transaction completion, the other party does not suffer in any manner. Another noteworthy contribution is that the protocol allows the customer to verify that the product he is about to receive is the one he actually ordered, before the customer pays for the product. At the same time it ensures that the customer receives the product if and only if the merchant gets paid for the product. All these features are achieved without significantly increasing the communication overhead or interactions with the third party as compared with similar protocols.

1. INTRODUCTION

In an electronic commerce environment the merchants and the customers are reluctant to trust each other and the following scenario is not uncommon. A customer is not willing to pay for a product without being sure it is the correct product sent by the merchant. A merchant is not willing to give the product unless he is sure that he will receive payment. If the merchant delivers the product without receiving the payment, the fraudulent customer may receive the product and then disappear without trace, causing loss for the merchant. If the customer pays before receiving the product, the merchant may not deliver or may deliver some wrong product. To address this problem we propose a fair exchange protocol that ensures the two parties get their respective items without allowing either party to gain an advantage by quitting or otherwise misbehaving.

Fair exchange protocols have been proposed in the context of electronic mails [2, 8] and electronic transactions [1, 3]. Most of these works [1,2, 8] focus on storing evidence that is to be used in case one party misbehaves. If a dispute occurs, a judge looks at the evidence and delivers his judgment. This dispute resolution is done after the protocol execution, that is, after the customer has obtained his product or the merchant his money. However, such "after-the-fact" protection [3,4] may be inadequate in an e-commerce environment where the customer and the merchant may not have identifiable places of existence and may be unreachable after the transaction. This motivates us to propose a protocol in which dispute resolution is within the scope of the protocol.

The e-commerce protocol that we develop is based on a theory of cross validation. A merchant has several products. He places a description of his products and the encrypted products in a catalog server. If the customer is interested in a product, he downloads the encrypted version of the product. When the customer agress to purchase the product, the merchant sends it encrypted with a second key such that this key bears a mathematical relation with the key the merchant used when putting up the encrypted product on the catalog server. The mathematical relation between the keys is such that the encrypted messages compare if and only if the unencrypted messages compare. Thus, by comparing the encrypted product received with the encrypted product that the customer downloaded from the catalog, the customer can be sure that the product he is about to pay for is indeed the product he wanted. Once the customer is satisfied with his comparison, he sends his payment token to a trusted third party. At the same time, the merchant sends the decrypting key to the third party. The third party verifies the customer's financial information and forwards the payment token to the merchant and the decrypting key to the customer. Thus we ensure that fairness is established in the protocol.

Tygar [7] has identified three desirable properties of a secure e-commerce protocol. These are the *money atomicity*, *goods atomicity* and *certified delivery* properties. To prevent any misbehavior during the execution of the protocol, we propose a new property which we call the *validated receipt* property. This property allows the customer to verify the contents of the product the merchant is about to deliver *before* making the payment. We reason that our protocol satisfies all of these properties.

The rest of the paper is organized as follows: Section 2 presents the theory for cross validation and then introduces the validated receipt property. This section also describes briefly the product validation process based on this validated receipt property. Section 3 describes the complete protocol. Section 4 shows informally that the protocol has all the desirable properties of secure e-commerce protocols. This section also discusses how transaction disputes are resolved automatically without human arbitration. Finally, Section 5 concludes the paper.

2. THEORY FOR CROSS VALIDATION

Before presenting our protocol we establish the theory of cross-validation on which the protocol is based. For lack of space we omit the proofs for the theorems presented here. The interested reader is referred to [6].

We assume that the item that needs to be validated by the customer, is transferred from the merchant to the customer in the form of a message. Examples of such products are digital library items such as, electronic papers, magazines, books, images, internet movies, music etc.

Definition 1 The set of *messages* $\mathcal{M}$ is the set of non negative integers m that are less than an upper bound N, i.e.

$$\mathcal{M} = \{m | 0 \leq m < N\} \tag{1}$$

Definition 2 For positive integers a, b and N, we say a is *equivalent* to b, modulo N, denoted by $a \equiv b \bmod n$, if $a \bmod n = b \bmod n$.

Definition 3 For positive integers a, x, n and $n > 1$, if $\gcd(a, n) = 1$ and $a.x \equiv 1 \bmod n$, then x is referred to as the *multiplicative inverse of a modulo n*. Two integers a, b are said to be *relatively prime* if their only common divisor is 1, that is, $\gcd(a, b) = 1$. The integers $n_1, n_2, \ldots, n_k$ are said to be *pairwise relatively prime*, if $\gcd(n_i, n_j) = 1$ for $i \neq j$.

Definition 4 The Euler's totient function $\phi(N)$ is defined as the number of integers that are less than N and relatively prime to N.

Theorem 1 Euler's theorem states that for every a and N that are relatively prime,

$$a^{\phi(n)} \equiv 1 \bmod N$$

Corollary 1 If $0 < m < N$ and $N = N_1 N_2 \ldots N_k$ and $N_1, N_2, \ldots, N_k$ are primes, then $m^{x\phi(N)+1} \equiv m \bmod N$.

Definition 5 A *key* K is defined to be the ordered pair $< e, N >$, where N is a product of distinct primes, $N \geq M$, where M is the largest message in the set of messages $\mathcal{M}$, and e is relatively prime to $\phi(N)$; e is the *exponent* and N is the *base* of the key K.

Definition 6 The *encryption* of a message m with the key $K = < e, N >$, denoted as $[m,K]$, is defined as

$$[m, < e, N >] = m^e \bmod N \tag{2}$$

Definition 7 The *inverse* of a key $K =< e, N >$, denoted by K^{-1}, is an ordered pair $< d, N >$, satisfying $ed \equiv 1 \bmod \phi(N)$.

Theorem 2 For any message m.

$$\left[[m, K], K^{-1}\right] = \left[\left[m, K^{-1}\right], K\right] = m \tag{3}$$

where $K =< e, N >$ and $K^{-1} =< d, N >$.

Corollary 2 An encryption, $[m, K]$, is *one-to-one* if it satisfies the relation

$$\left[[m, K], K^{-1}\right] = \left[\left[m, K^{-1}\right], K\right] = m$$

Definition 8 Two keys $K_1 =< e_1, N_1 >$ and $K_2 =< e_2, N_2 >$ are said to be *compatible* if $e_1 = e_2$ and N_1 and N_2 are relatively prime.

Definition 9 If two keys $K_1 =< e, N_1 >$ and $K_2 =< e, N_2 >$ are *compatible*, then the *product* key, $K_1 \times K_2$, is defined as $< e, N_1 N_2 >$.

Lemma 1 For positive integers a, N_1 and N_2,

$$(a \bmod N_1 N_2) \equiv a \bmod N_1$$

Theorems For any two messages m and $\hat{m}$, such that $m, \hat{m} < N_1, N_2$,

$$[m, K_1 \times K_2] \equiv [\hat{m}, K_1] \bmod N_1 \text{ if and only if } m = \hat{m} \tag{4}$$

$$[m, K_1 \times K_2] \equiv [\hat{m}, K_2] \bmod N_2 \text{ if and only if } m = \hat{m} \tag{5}$$

where K_1 is the key $< e, N_1 >$, K_2 is the key $< e, N_2 >$ and $K_1 \times K_2$ is the product key $< e, N_1 N_2 >$.

2.1. VALIDATED RECEIPT PROPERTY

The validated receipt property is stated as follows:

Validated Receipt A customer is able to ensure that the product he is about to receive from the merchant is the same as the product he ordered, before the customer pays for the product.

Our protocol achieves the validated receipt property using the results of theorem 3. Let m be the product to be delivered. The values N_1 and N_2 are public. The merchant generates the set of keys (K_1, K_1^{-1}), and sends m, K_1 and K_1^{-1} to a trusted third party. The trusted third party computes $T = [m, K_1]$ (that is encrypts m with the key K_1) and places T at a public place, henceforth called the catalog, as an advertisement for m. When the customer decides to purchase m from the merchant, the customer acquires T from the catalog and keeps it for future validation of the product received.

To sell m to the customer, the merchant selects a second set of keys (K_2, K_2^{-1}) such that K_2 is compatible with K_1 according to definition 8. The merchant escrows the key K_2^{-1} with the trusted third party and provides the customer with $C = [m, K_1 \times K_2]$.

The customer verifies that T and C are encryption of the same message m by verifying: $T \equiv C \bmod N_1$, as per equation (4)

When satisfied, the customer requests the key K_2^{-1} from the trusted third party and decrypts C to obtain m using

$$m = \left[C, K_2^{-1}\right]$$

The proof of correctness follows from theorem 3:

$$[m, K_1 \times K_2] \equiv [\hat{m}, K_2] \bmod N_2 \text{ if and only if } m = \hat{m}$$

3. THE COMPLETE PROTOCOL

3.1. ASSUMPTIONS

We make the following assumptions in the protocol:

1 We assume the existence of an on-line trusted third party.

2 Before the protocol is initiated, mutual authentication takes place between the trusted third party, the customer and the merchant and secure channels are set up between them. All communications are assumed to occur over these secure channels so that confidentiality of messages in transit is ensured. Note, we do not assume that integrity of messages will be ensured; nor do we assume that the secure channels are immune to replay attacks.

3 We assume that a message transmitted over a channel is guaranteed to be delivered.

4 We assume that all encryptions are strong enough that the receiver of an encrypted message is unable to decrypt the message without the appropriate key.

5 All parties use the same algorithm for encryption as well as for generating cryptographic checksums.

6 Financial institutions are assumed to be trusted. The customer and its financial institution shares a secret key that was established when the customer opened an account with the financial institution.

7 Payment for product is in the form of a token, $\mathcal{P}$, that is accepted by the merchant.

8 The merchant advertises the product with the trusted third party by keeping an encrypted copy of the product, $[m,K_1]$, with the trusted third party, together with a description of the product. Note that the merchant actually creates the key pair K_1 and K_1^{-1} and sends m, K_1 and K_1^{-1} to the trusted party. The trusted third party performs the encryption before advertising the product on the catalog. We prefer this approach over the merchant providing the encrypted copy of the product, because, in this manner, the trusted third party is able to certify that the product meets its claims.

Table 3.1 lists the notations used in the description of the protocol.

3.2. PHASE 1: INTENT TO PURCHASE PRODUCT

1 $\mathbf{TP} \Longrightarrow \mathbf{C}$: PID, $[m,K_1]$

2 C $\Longrightarrow$ M: PO= {purchase-order,[CC(purchase-order),C_{prv}]},
where purchase-order = {PID,C,M,Agreed_Price, ψ_C}

Message 1 The customer browses the product catalog located at the trusted third party, and chooses the product m he wants to buy. Then he gets the encrypted form of the product, namely, $[m,K_1]$, together with the product identifier, PID.
Message 2 The customer decides on a price (Agreed_Price) to pay for the product and prepares a purchase order. The purchase order contains the following information:

(i) the product identifier, PID

(ii) the customer's identity, C

Table 3.1 . Symbols used in protocol description

C, M and TP	Ids for customer, merchant and trusted third party
A_{prv}, A_{pub}	A's private and public keys
A $\Longrightarrow$ B: X	A sends X to B
m	Product the customer purchases
PID	The id for product m
C_{Act}	Customer's account information with customer's financial institution
CF	A secret key shared between the customer and its financial institution
F_C	An Id for customer's financial institution
$\mathcal{P}$	Payment token used for paying for goods
[X,K]	encryption of X with key K
CC(X)	A cryptographic checksum of X, using an algorithm such as the Secure Hash [5]
K^{-1}	decryption key corresponding to encryption key K
ψ_A	a nonce for entity A. Each entity's nonces are unique and different from other entities' nonces.

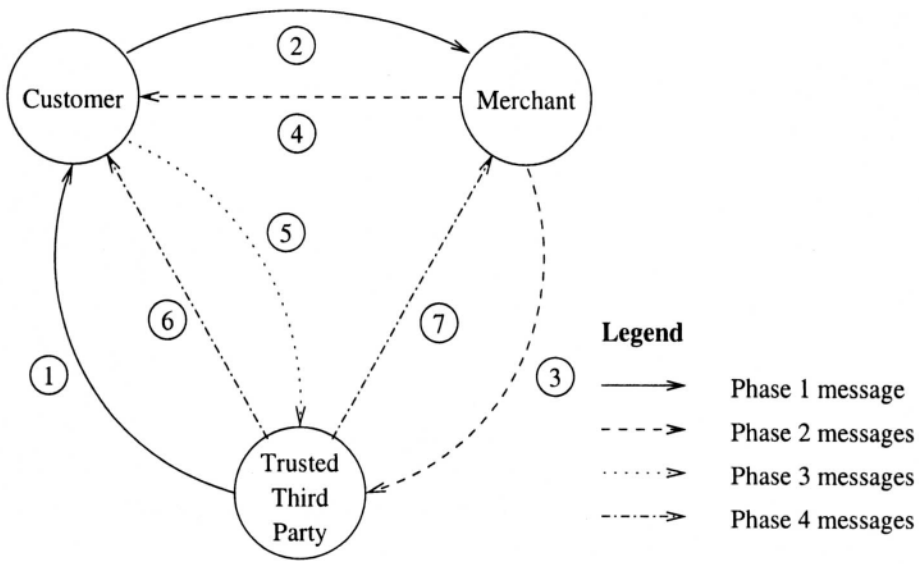

Figure 3.1. Messages exchanged in the e-commerce protocol

(iii) the identity of the merchant, M

(iv) the price of the product, Agreed_Price, and

(v) a nonce, ψ_C, from the customer.

The customer generates a cryptographic checksum of the purchase-order and then digitally signs the digest. The cryptographic checksum of the purchase-order forestalls debate over the details of the order, or whether the order was received completely and correctly. The customer's signature forestalls debate over whether the customer expressed intention to purchase the product. The nonce, ψ_C, in the purchase order forestalls a replay of the purchase order with the merchant.

The purchase order and its signed checksum, together henceforth called PO, is then forwarded to the merchant.

3.3. PHASE 2: KEY ESCROW AND PRODUCT DELIVERY

3 $\mathbf{M} \Longrightarrow$ TP: $[[\text{CC(purchase-order)},\mathbf{C}_{prv}],\mathbf{M}_{prv}], K_2^{-1}$, $[\text{CC}([m,K_1 \times K_2]),\mathbf{M}_{prv}]$

4 $\mathbf{M} \Longrightarrow C$: $[m,K_1 \times K_2]$, $[\text{CC}([m,K_1 \times K_2]),\mathbf{M}_{prv}]$

Message 3 The merchant endorses the purchase order received from the customer, provided the merchant agrees to all its contents (that is, the Agreed_Price is indeed the price agreed upon for the product *m*); the merchant then digitally signs the cryptographic checksum of the purchase-order bearing the customer's signature, that is [CC(purchase-order), $\mathbf{C}_{prv}$] and forwards it to the trusted third party. This prevents the merchant claiming later on that he hadn't agreed to the terms and conditions of the transaction.

The merchant, at this time, generates a second set of keys K_2 and K_2^{-1} such that K_1 and K_2 are compatible. He encrypts *m* with the product key, $K_1 \times K_2$ and prepares a cryptographic checksum, $\text{CC}([m,K_1 \times K_2])$, from it. The merchant digitally signs this digest and forwards it, together with the key K_2^{-1}, to the trusted third party. The signed digest for $[m,K_1 \times K_2]$ provides certified delivery.

Message 4 To the customer the merchant sends the encrypted product $[m,K_1 \times K_2]$ together with its signed cryptographic checksum $[\text{CC}([m,K_1 \times K_2]),\mathbf{M}_{prv}]$. The signed cryptographic checksum establishes origin of the product and also forestalls debate over the product being corrupted in transit.

3.4. PHASE 3: PRODUCT VALIDATION AND PAYMENT FOR PRODUCT

5 $\mathbf{C} \Longrightarrow$ TP: PO, $[\text{CC}([m, K_1 \times K_2]),\mathbf{C}_{prv}]$, $[\mathcal{P},\mathbf{C}_{prv}]$, $[\text{CC}(\mathcal{P}),\mathbf{C}_{prv}]$, where $\mathcal{P} = \{\mathbf{F}_C,\text{C},[\mathbf{C}_{Act},\text{CF}],\text{Agreed\_Price},\psi_C\}$

Message 5 The customer validates the product by comparing $[m,K_1]$ with $[m,K_1 \times K_2]$ (as outlined in Section 2.1). If the two compare, the customer requests the decrypting key, K_2^{-1}, from the trusted third party. To do this, the customer forwards to the trusted third party, PO (as generated in Phase 1 above), signed payment token, $\mathcal{P}$ together with its cryptographic checksum, and a signed cryptographic checksum of the encrypted product received, $[m, K_1 \times K_2]$.

The payment token contains the following information:

(i) the identity of the customer's financial institution, $\mathbf{F_C}$

(ii) the customer's identity, C

(iii) the customer's account information with the financial institution, $\mathbf{C_{Act}}$

(iv) the amount to be debited from the customer's account, Agreed_Price and

(v) a nonce of the customer, ψ_C.

The customer's account information is encrypted with the secret key, CF, shared between the customer and his financial institution. This ensures that nobody other than the customer and his financial institution can access this information. The nonce, ψ_C, in the payment token ensures that it is not susceptible to replay attacks. The customer prepares a digest of the payment token, $\mathbf{CC}(\mathcal{P})$ and then digitally signs the token and the digest. The digest forestalls debate over the contents of the payment token and the customer's signature forestalls debate by customer regarding amount debited from his account.

The signed cryptographic checksum of the product received, $[m,K_1 \times K_2]$ ensures certified delivery.

3.5. PHASE 4: PRODUCT AND PAYMENT RECEIPT

6 $\mathbf{TP} \Longrightarrow \mathbf{C}: [K_2^{-1}, \mathrm{TP}_{prv}]$

7 $\mathbf{TP} \Longrightarrow \mathbf{M}: [\mathcal{P}, \mathrm{TP}_{prv}]$

Messages 6 and 7 The trusted third party first compares the digest included in PO from the customer (received in Message 5), with the digest of the same from the merchant (as received in Message 3). If the two do not compare the trusted third party aborts the transaction. If they do, the trusted third party next validates the payment token with the customer's financial institution by presenting the token as well as the agreed upon sale price, Agreed_Price (from the purchase-order). The financial institution validates the token only if the two prices (the one from the payment token and the one supplied by the trusted third party) match and the customer has sufficient funds in his account for the payment. If the token is not validated, the trusted third party aborts the protocol

by informing the merchant about this. If token is validated, the trusted third party sends the decrypting key K_2^{-1} to the customer and the payment token $\mathcal{P}$ to the merchant, both digitally signed with the trusted third party's private key.

4. PROTOCOL ANALYSIS

The e-commerce protocol presented here satisfies all the desirable properties of secure e-commerce protocols. Secure channels guarantee the confidentiality of all messages. Transmission freshness of request and/or response is guaranteed by including nonces within the relevant messages. Non-repudiation of the origin for the request and/or response is provided because, wherever required, such requests and/or responses are digitally signed by the sender's private keys.

The protocol ensures money atomicity as follows: The payment token generated by the customer contains the amount to be debited from the customer's account and credited to the merchants account. Consequently no money is created or destroyed in the system (comprising of the merchant's account and the customer's account) by this protocol. Moreover, the nonces in the payment token ensure that the merchant cannot debit the customer's account multiple times for the same purchase.

Goods atomicity is ensured because the trusted third party hands over the payment token only when the customer acknowledges the receipt of the product; the protocol also ensures that the product is actually available to the customer for use, only when the customer gives the go-ahead for payment (by acknowledging the receipt of the good).

Certified delivery is achieved as follows. The trusted third party receives a cryptographic checksum of the product from the merchant. Also the customer independently generates a checksum of the product received and sends it to the trusted third party. Using these two copies of the cryptographic checksums available at the trusted third party both the merchant and the consumer are able to give non-repudiable proof of the contents of the delivered goods.

Finally validated receipt is ensured in the protocol. This has been illustrated earlier in section 2.1.

4.1. DISPUTE HANDLING

Our e-commerce protocol, is able to handle almost all possible dispute scenarios without human arbitration.

Customer complains that product is not as advertised Such a complaint is prevented in the protocol because the trusted third party is involved in advertising the product on the catalog. Recall that the trusted third party receives m, K_1 and K_1^{-1} from the merchant together with a description of m. The trusted third party compares m with its description before encrypting m with key K_1 and placing it on the catalog.

Customer complains about incorrect or damaged product The validated receipt property ensures that the customer requests the decryption key, K_2^{-1} only after the customer is satisfied that the product received is correct. Consequently, if such a complaint is ever made, it is not entertained.

Customer complains about incorrect decryption key K_2^{-1} The trusted third party takes the following steps:

1 From the copy, $[m, K_1]$ that the trusted third party has on the catalog, it gets the product m and sends it to the customer.

2 The trusted third party may optionally take appropriate action with the merchant to prevent such problem/fraud in future.

Customer complains that he was charged more than what he agreed to The trusted third party has a copy of the purchase order, PO, signed by the customer and hence a proof of what the customer agreed to pay. Consequently, such a claim is not entertained.

Customer complains that he has been wrongly charged The trusted third part; can settle this dispute by producing the signed purchase order.

Merchant complains of inadequate payment Such a claim is not entertained because the trusted third party validates the payment token with the customer's financial institution.

Merchant complains that payment token was not received The trusted third party re-delivers the payment token. Note that even if the merchant receives the payment token multiple times, it can be used only once because of the presence of the customer's nonce in the payment token.

5. CONCLUSION AND FUTURE WORK

In this work we have proposed a new e-commerce protocol for performing business over the Internet. The protocol relies on an online trusted third party. An important feature of this protocol is that it tries to avoid disputes between the transacting parties. If disputes still arise, the protocol can handle these automatically, without manual intervention and within the protocol itself. The protocol allows the customer to be confident that he is paying for the correct product before actually paying for it. The protocol also ensures that the customer does not get the product unless he pays for it and that the merchant does not get paid unless he delivers the product.

A major bottleneck in the protocol is the trusted third party. Not only is the performance of the trusted third party an issue, but also its vulnerability to denial of service attacks. However, this is not a problem which is limited to our protocol. This bottleneck is present in all e-commerce protocols that

require a trusted third party for their operation. We are currently investigating two approaches to reduce this problem. In the first approach we are looking at ways to modify the protocol to reduce the interactions with the trusted third party. In the second approach we are looking at the multiple roles played by the trusted third party and ways to distribute these roles over a number of (possibly) semi-trusted third parties. This second approach will also help in making our protocol fault-tolerant.

Acknowledgment

The works of Indrajit Ray and Indrakshi Ray were partially supported by the NSF under grant EIA 9977548 and by a Faculty Research Grant from the University of Michigan-Dearborn.

References

[1] B. Cox, J. D. Tygar, and M. Sirbu. NetBill Security and Transaction Protocol. In *Proceedings of the First USENIX Workshop in Electronic Commerce*, pages 77–88, July 1995.

[2] 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.

[3] S. Ketchpel. Transaction Protection for Information Buyers and Sellers. In *Proceedings of the Dartmouth Institute for Advanced Graduate Studies '95: Electronic Publishing and the Information Superhighway, 1995*, 1995.

[4] S. Ketchpel and H. Garcia-Molina. Making Trust Explicit in Distributed Commerce Transactions. In *Proceedings of the Sixteenth International Conference on Distributed Computing Systems*, pages 270–281,1996.

[5] National Institute of Standards. FIPS 180: Secure Hash Standard, April 1993. Federal Information Processing Standard.

[6] I. Ray, I. Ray, and N. Narasimhamurthi. A Fair-exchange E-commerce Protocol with Automated Dispute Resolution. Technical Report CIS-TR-010-00, Computer and Information Science Department, University of Michigan-Dearborn, 2000.

[7] J. D. Tygar. Atomicity in Electronic Commerce. In *Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing*, pages 8–26, May 1996.

[8] J. Zhou and D. Gollmann. A Fair Non-repudiation Protocol. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 55–61, Oakland, California, May 1996.