# A Fair Non-Repudiation Protocol

Bo Meng
*College of Computer Science and Technology*
*Wuhan University of Technology*
*Wuhan 430063 P. R. China*
*mengbo@263.net.cn*

Shaomei Wang
*Port Machinery CAD/CAE Center*
*Wuhan University of Technology*
*Wuhan 430063 P. R. China*
*sunwang@public.wh.hb.cn*

Qianxing Xiong
*College of Computer Science and Technology*
*Wuhan University of Technology*
*Wuhan 430063 P. R. China*
*qxxi@public.wh.hb.cn*

## Abstract

*Non-repudiation and fairness are two important issues in the implementation of electronic commerce. This paper discusses non-repudiation services and fair non-repudiation protocols. After analyzing two widely used fair non-repudiation protocols the Fair Non-Repudiation Protocol and the Certified Mail Protocol, we propose a new Fair Non-Repudiation Protocol, which transports less data, has better confidentiality of message and better fairness. And we analyze it with accountability framework on its non-repudiation and fairness.*

## 1. Introduction

As electronic commerce comes into practice, non-repudiation and fairness become two important issues. In an electronic commerce transaction, it is essential to guarantee both the non-repudiation for the actions of each party involved in the transaction and the fairness for each one [1,2,3]. The non-repudiation and fairness in electronic commerce is achieved by applying fair non-repudiation protocols.

A non-repudiation protocol specifies what kinds of non-repudiation services it provides and how to implement them [4,5,6]. The purpose of non-repudiation services is to generate, collect, and maintain the evidences on the events and actions and to protect the parties involved in a transaction against the other party denying that a particular event or action took place. It generates the irrefutable evidence to support the resolution of any such disagreement [7].

The non-repudiation services usually used are the Non-Repudiation of Origin (NRO), Non-Repudiation of Delivery (NRD), Non-Repudiation of Submission (NRS) ,and Non-Repudiation of Transport (NRT).

Non-repudiation of origin provides the recipient evidence that will protect against any attempt by the originator to falsely deny having sent the message. Non-repudiation of delivery provides the originator evidence that will protect against any attempt by the intended recipient to falsely deny having received the message. Non-repudiation of submission provides the originator evidence that will protect against any attempt by the delivery agency to falsely deny having submitted the message. Non-repudiation of transport provides the originator evidence that will protect against any attempt by the transporter to falsely deny having transported the message.

The mechanism of non-repudiation concerns the exchange of the special non-repudiation tokens. A non-repudiation token consists of digital signature and data appendix. The non-repudiation token is stored as evidences and used to resolve disputations.

The non-repudiation service is composed of four phases: evidence generation, evidence transport and storage, evidence verification, and disputation resolution [8].

A fair non-repudiation protocol should be that a proper execution of the protocol ensures that the Non-Repudiation of Delivery Token (NRDT) and the Non-Repudiation of Origin Token (NROT) are available to both the originator and the intended recipient respectively. Moreover, the protocol must be fail-safe [1]. That is to say, an incomplete execution of the protocol will not result in a situation where the NRDT is available to the originator but the NROT is not available to the intended recipient, or vice versa.

## 2. Related protocols

The non-repudiation protocols are classified into two classes with whether they use the Trusted Third Party (TTP). One is based on the trusted third party. The other is not. A lot of non-repudiation protocols are based on the trusted third party, such as the Fair Non-repudiation Protocol (FNP) proposed by J. Zhou in [2], the Certified Mail Protocol (CMP) proposed by Robert H. Deng in [1], the Certified Electronic Mail (CEM) protocol proposed by

A. Bahreman in [9], the Non-repudiation with Mandatory NRR (NMNRR) protocol proposed by T. Coffery in [10] and so on. In the following we analyze the FNP and CMP in detail.

## 2.1. FNP

The purpose of FNP is to provide the originator and the intended recipient with evidences after an execution of the protocol without giving any party an advantage during the execution. After the execution of the protocol, the following evidences are generated: evidence of origin, evidence of delivery, evidence of submission, and evidence of confirmation.

The protocol is outlined as follows: The message to be sent consists of tow parts. One is encrypted text *C*; the other is a key *K*. The sender *A* sends his digital signature and the encrypted text *C* to the intended recipient *B*. *B* verifies *A*'s digital signature and acknowledges the receipt of *C*. When *A* receives the acknowledgement from *B*, it verifies the digital signature before proceeding. *A* then submits *K* to *TTP*. *TTP* places *K* in a place where *B* can get it. When *B* gets *K*, he can decrypt the encrypted text *C* and obtain the message *A* sent. The protocol is described informally as following:

*M*□message sent from *A* to *B*
*C*□ commitment for message *M* encrypted under key *K*

*K*□message key
*L*□ a unique label to identify *M*
$EOO=$□$sS_A(feoo,B,L,C)$ evidence of origin of *C*
$EOR=$□$sS_B(feor,A,L,C)$ evidence of receipt of *C*
$Subk=$□$sS_A(fsub,B,L,K)$ evidence of submission of *K*
$Conk=: sS_{TTP}(fcon,A,B,L,K)$ evidence of comfirmation of *K*

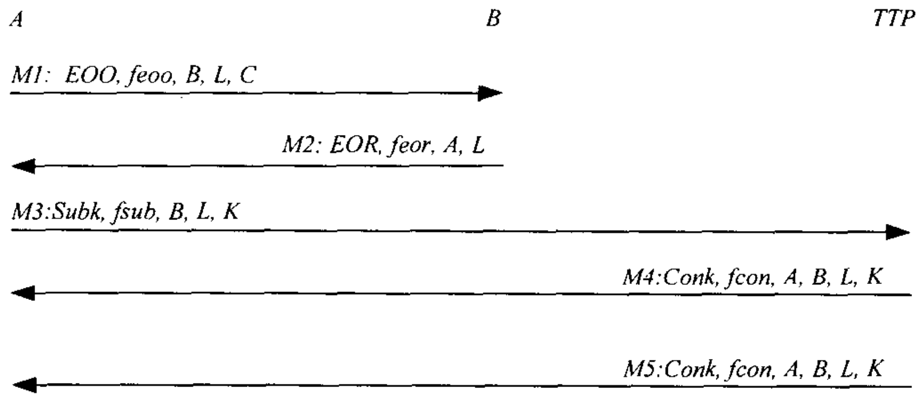

Figure 1. FNP

The proposer of the protocol thinks the protocol is fair. But it is not fair according to our definition of fairness. This is because if *B* gives up after *B* finishes the first step, *B* does not know the content of the message, but he gets the NROT. Besides, the protocol is designed to transport more pieces of messages than CMP when running and include *C* in the evidence, which increase the amount of data transport.

## 2.2. CMP

CMP has two variant models: CMP1 and CMP2. We discuss CMP1 here. After an execution of CMP1, following evidences are generated: evidence of origin (*POO_PM*) and evidence of delivery (*POD_PM*). There are three active parties involved in CMP1: the mail originator *A*, the mail recipient *B*, and the postman *PM*.

*m* - message sent from *A* to *B*
$PK_A$ - public key of *A*
$SK_A$ private key of *A*
$\{m\}PK_A$: encryption of message *m* under $PK_A$
$\{m\}SK_A$□message *m* signed with $SK_A$
$[m]k$: encryption of message *m* under symmetric-key *k*

An execution of CMP1(Figure 2) is as following:

The mail originator *A* constructs and sends *M1* to the mail recipient *B*. After receiving *M1*, *B* decides whether run the protocol or not. If *B* decides to run the protocol, it constructs and sends *M2* to *PM*. When *PM* receives *M2*, it verifies $\{A, B, PM, h(m)\}SK_B$, gets *k* by decrypting $\{k\}PK_{PM}$ with $SK_{PM}$, gets $\{A, B, PM, m\}SK_A$ by decrypting $[\{A, B, PM, m\}SK_A]k$ with *k* and verifies it, Compare *h(m)* gotten from *A* with *h(m)* gotten from *B*. If the two values match, *PM* constructs *POO_PM* and *POD_PM* and sends them. That ends the execution of the protocol.

In an execution of the protocol the message need to be transmitted four times. Although the protocol needs to transport few messages than FNP, the efficiency is lower when the message is very large.

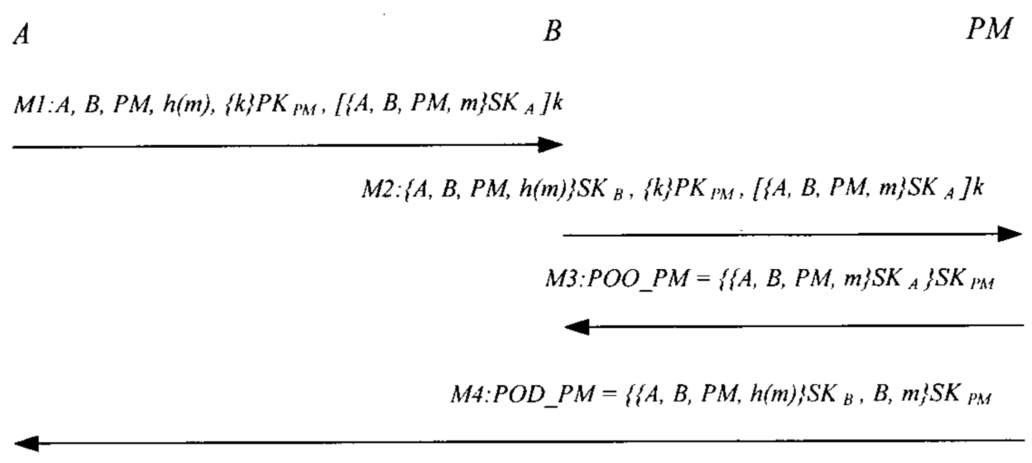
A
B
PM
M1:A, B, PM, h(m), {k}PK_PM, [{A, B, PM, m}SK_A]k
M2:{A, B, PM, h(m)}SK_B, {k}PK_PM, [{A, B, PM, m}SK_A]k
M3:POO_PM = {{A, B, PM, m}SK_A}SK_PM
M4:POD_PM = {{A, B, PM, h(m)}SK_B, B, m}SK_PM

Figure 2. CMP1

## 3. Fair non-repudiation protocol

In this paper, we propose a new fair non-repudiation protocol FNRP. It generates NROT and NRDT and is based on TTP, while the TTP here is not the Delivery Agency but an organization to generate NROT and NRDT. $A$ retrieves the NRDT and $B$ retrieves NROT, which does the same way as in FNP. In order to short message, digest of ciphertext of message is used in both the NROT and the NRDT instead of plaintext of message. Therefore, TTP will not know the content of message, and this is good for both $A$ and $B$.

The FNRP can be described as following:

$m$: message sent from $A$ to $B$

$A$: identifier of $m$ of originator

$B$: identifier of $m$ of the intended recipient

$TTP$: identifier of $TTP$

$h(m)$: one-way hash function of $m$

$ID$: identifier of $m$

$Tg$: time and date of token generated by $TTP$

$T$: time and date of sending message

$PK_A$: public key of $A$

$SK_A$ private key of $A$

$\{m\}SK_A$ digital signature of $m$ under $SK_A$

$k$: symmetric key $k$

$C=e(m,k)$: $C$ is ciphertext of $m$ under $k$

$\{m\}PK_A$: encryption of message $m$ under $PK_A$

$NROT=Tg \| Z_1 \| \{Z_1\} SK_A \|$
$\{Tg \| Z_1 \| \{Z_1\} SK_A\} SK_{TTP}$,

$Z_1= k \| A \| B \| TTP \| T_1 \| ID \| h(C)$

$NRDT= B \| k \| Tg \| Z_2 \| \{Z_2\} SK_B \|$
$\{\| B \| k \| Tg \| Z_2 \| \{Z_2\} SK_B\} SK_{TTP}$,

$Z_2= A \| B \| TTP \| T_2 \| ID \| h(C)$

$A$ generates $C$ and $h(C)$ of the message to be sent, encrypts $k$ with $PK_{TTP}$, generates the digital signature of $Z_1$, and then sends $M1$ to $B$. $B$ decides whether run the protocol or not after receiving $M1$. If $B$ decides to abort, it will neither know the content of the message nor get $A$'s NROT, and this is fair to both parties. If $B$ agrees to run the protocol, it will generate $h(C)$ and construct $Z_2$, and then generate $\{Z_2\} SK_B$ and $M2$, finally send $M2$ to $TTP$. When $TTP$ receives $M2$, it verifies $B$'s digital signature, gets $k$ by decrypting $\{k\}PK_{TTP}$, decrypts $e(Z_1 \| \{Z_1\} SK_A,k)$, and verifies $A$'s digital signature. $TTP$ compares the $k$ and $h(C)$ getting from $e(Z_1 \| \{Z_1\} SK_A,k)$, which is also from $A$, with the $k$ getting from $\{k\}PK_{TTP}$ and $h(C)$ getting from $Z_2 \| \{Z_2\} SK_B$ respectively . If both two values of $k$ match and two values of $h(C)$ match, $TTP$ generates NROT retrieved by $B$ and $NRDT$ retrieved by $A$; otherwise, terminates the running of the protocol, which is fair to both $A$ and $B$ since $B$ does not get NROT and $A$ does not get NRDT. At the same time $B$ does not know the content of the message either.

The tokens NROT and NRDT generated by the FNRP conform to the ISO/IEC 13888, so arbitrators can use them as evidences to resolve the disputations.

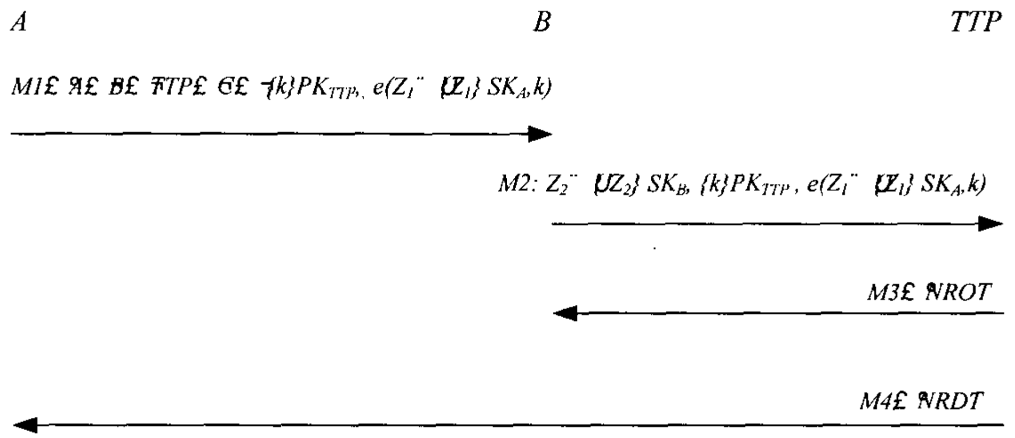
A
B
TTP
M1£ A£ B£ TTP£ C£ {k}PK_TTP, e(Z_1¨ {Z_1} SK_A,k)
M2: Z_2¨ {Z_2} SK_B, {k}PK_TTP, e(Z_1¨ {Z_1} SK_A,k)
M3£ NROT
M4£ NRDT

Figure 3. FNRP

## 4. Analysis of FNRP

### 4.1. Accountability analysis

In this subsection we analyze FNRP using the accountability framework proposed in [11]. The main objective of this analysis is to prove that FNRP satisfies the non-repudiation of origin and non-repudiation of delivery requirements. Owning to the space, only the constructs and postulates used in our analysis are listed.

**Constructs:**

*A CanProve x*

Principal $A$ can prove statement $x$ if, for any principal $B$, $A$ can execute a sequence of operations such that after the sequence of operations, it has proved $x$ to $B$ without revealing any secret $y$ ($y \neq x$) to $B$.

*PK Authenticates A*

This construct is used to denote the fact that key $PK$ can be used to authenticate the signature of principal $A$, or to associate $A$ unambiguously with any statement digitally signed with $SK$.

*x in m*

$x$ is an interpretation of a field, or an interpretation of a combination of fields in message $m$. This interpretation is protocol specific, and expected to be defined by the protocol designers explicitly.

*A Says x*

Principal $A$ is accountable for making the statement $x$, and anything that $x$ implies. It is used for interpreting provability results of the form *A CanProve (B Says x)*. Note that a principal. $A$ cannot prove that *B Says x* if $x$ is a signed message field that is encrypted using a shared key. That is, if $x$ has been encrypted for confidentiality using a key shared between $A$ and $B$, then $A$ cannot prove that *B Says x*. This follows from the definition of the *CanProve* construct, which says that in order to prove accountability, the prover should not have to reveal secrets (which are different from $x$ to the audience.)

*A Receives m SignedWith SK*

Principal $A$ receives message $m$ which is signed with $SK$.

*A IsTrustedOn x*

Principal $A$ is trusted on statement $x$. In particular, $A$ has the authority to endorse statement $x$, and is liable for making statement $x$.

**Postulates:**

**Conjunction:**

$$\frac{A\ CanProve\ x;\ A\ CanProve\ y}{A\ CanProve\ (x \wedge y)}$$

That is, if $A$ can prove that $x$ holds and $A$ can prove that $y$ holds, then $A$ can prove that $x \wedge y$ holds.

**Inference:**

$$\frac{A\ CanProve\ x;\ x \Rightarrow y}{A\ CanProve\ y}$$

That is, if $A$ can prove that $x$ holds, and if $x$ implies y, then it follows that $y$ holds. Here the statement $x => y$ ($x$ implies $y$) is used to articulate the interpretations of signed messages. Such interpretations are assumed to be defined explicitly by protocol designers, and hence, are assumed to be evident to all principals involved.

**Signature:**

$$\frac{A\ Receives\ (m\ SignedWith\ SK);\ x\ in\ m;\quad A\ CanProve\ (PK\ Authenticates\ B)}{A\ CanProve\ (B\ Says\ x);}$$

That is, if $A$ receives a message $m$ that is signed with key $SK$, the message $m$ contains statement $x$, and if $A$ can prove that the key $PK$ authenticates principal $B$ or that $PK$ authenticated $B$ at the time the message was signed, then $A$ can prove that $B$ indeed says $x$.

**Trust:**

*A CanProve (B Says x);*

*A CanProve (B IsTrustedOn x)*

---

*A CanProve x ;*

That is, if *A* can prove that *B*, who is an authority on *x*, says *x*, then *A* can prove that *x* holds.

Using the constructs of the accountability framework, the objective of the protocol is to generates the following goals:

*G1□A CanProve ( B Receives C* $\wedge$ *B Receives k )*

*G2: B CanProve ( A sent C* $\wedge$ *A sent k )*

First, we need to interpret the protocol messages (see Figure 3) using the constructs of the accountability framework. In the analysis, only those messages, which are signed and have plaintext contents, have accountability, and need be interpreted. Therefore, protocol message 1 will not be interpreted. Protocol messages 2, 3, and 4 are interpreted as:

*M2□TTP Receives (h(C) SignedWith* $SK_B$*)*
*TTP Receives (k SignedWith* $SK_A$*)*

*M3: B Receives ((h(C) SignedWith* $SK_A$*) SignedWith* $SK_{TTP}$*)*
*B Receives ((k SignedWith* $SK_A$*) SignedWith* $SK_{TTP}$*)*

*M4: A Receives ((h(C) SignedWith* $SK_B$*) SignedWith* $SK_{TTP}$*)*
*A Receives ((*$\|B\|k\|$*) SignedWith* $SK_{TTP}$*)*

The initial assumptions that are required in the analysis are listed below:

*A1□A□B CanProve (*$PK_{TTP}$ *Authenticates TTP)*
*A2□A□TTP CanProve (*$PK_B$ *Authenticates B)*
*A3: B□TTP CanProve (*$PK_A$ *Authenticates A)*
*A4: A□B CanProve (TTP IsTrustOn (TTP Says))*
*A5: A Says (h(C))* $\Rightarrow$ *A sent C*
*A6: B Says (h(C))* $\Rightarrow$ *B Receives C*
*A7: A Says (k)* $\Rightarrow$ *A sent k*
*A8: TTP says (*$\|B\|\ k\|$*)* $\Rightarrow$ *B Receives k*

*M2* is equivalent to:

*M2.1□TTP Receives (h(C ) SignedWith* $SK_B$ *)*
*M2.2□TTP Receives (k SignedWith* $SK_A$ *)*

*When TTP received M2,*

1. *TTP CanProve (B Says h(C))* — *M2.1□A2□ Signature*
2. *TTP CanProve (B Received C)* — A6, *Inference*

This is because according to the protocol, only after receiving C, can *B* generate *h(C)*.

3. *TTP CanProve (A Says k)* — *M2.2, A3 Signature*
4. *TTP CanProve (A sent k)* — A7, *Inference*

*TTP* compares *k and h(C)* sent by *B* with *k and h(C)* sent by *A*. if either of two does not match, the execution of the protocol will be terminated; Otherwise, *TTP* will infer that *A* generated and sent *C* and *B* received *C*.

When *B* receives *M3*, *M3* can be interpreted as:

*M3.1 B Receives (h(C) SignedWith* $SK_A$*) SignedWith* $SK_{TTP}$
*M3.2 B Receives (k SignedWith* $SK_A$ *) SignedWith* $SK_{TTP}$

1. *B CanProve (TTP Says( h(C) SignedWith* $SK_A$*))* — *M3.1, A1, Signature*
2. *B CanProve (h(C) SignedWith* $SK_A$*)* — *A4, Trust*
3. *B CanProve □A Says h(C) □* — *A3□Signature*
4. *B CanProve (A sent C)* — *A5, inference*
5. *B CanProve (TTP Says (k SignedWith* $SK_A$*))* — *M3.2□A1, Signature*
6. *B CanProve (k SignedWith* $SK_A$*)* — *A4, Trust*
7. *B CanProve □A Says k□* — *A3□Signature*
8. *B CanProve (A sent k)* — *A7, inference*
9. *B CanProve ( A sent C* $\wedge$ *A sent k )* — *4, 8, Conjunction*

So far, *B* can prove that *A* sent *C* and *k*. According to the protocol, *B* can further prove that *A* sent *m*.

When *A* receives *M4*, *M4* can be interpreted as:

*M4.1 A Receives ((h(C) SignedWith* $SK_B$*) SignedWith* $SK_{TTP}$*)*
*M4.2 A Receives ((*$\|B\|k\|$*) SignedWith* $SK_{TTP}$*)*

1. *A CanProve (TTP Says ((h(C) SignedWith* $SK_B$*))* — *M4.1, A1, Signature*
2. *A CanProve ((h(C) SignedWith* $SK_B$*)* — *A4, Trust*
3. *A CanProve □B Says h(C) □* — *A2□Signature*
4. *A CanProve (B Receives C)* — *A6, Inference*
5. *A CanProve (TTP Says (*$\|B\|k\|$*))* — *M4.2, A1, Signature*
6. *A CanProve (B Receives k )* — *A8, Inference*
7. *A CanProve ( B Receives C* $\wedge$ *B Receives k )* — *4, 6, Conjunction*

So far, *A* can prove that *B* received *C* and *k*. According to the protocol, *A* can further prove that *B* got *m*.

## 4.2. Fairness analysis

A fair Non-repudiation protocol should be that proper execution of the protocol ensures that the non-repudiation of delivery token (NRDT) and the non-repudiation of origin token (NROT) are available to the intended recipient and originator, respectively. Moreover, the protocol must be fail-safe. That is to say, incomplete execution of the protocol will not result in a situation where the NRDT is available to the originator but the NROT is not available to the recipient, or vice versa.

In FNRP, when *A* sends *M1* to *B*, *B* has the choice of whether allow the procedure to go on. If *B* decides to abort, neither *A* nor *B* can get its non-repudiation token and *B* cannot know the content of the message, hence it is also fair to *A*. If *B* agrees to go on, it generates *M2* and sends it to *TTP*. *TTP* does what the protocol defined. *TTP* compares the *k* and *h(C)* getting from $e\ (Z_1 \| \{Z_1\}\ SK_A,\ k)$, which is also from A, with the *k* getting from $\{k\}PK_{TTP}$ and *h(C)* getting from $Z_2 \| \{\ Z_2\}\ SK_B$, respectively . If both two values of *k* match and two values of *h(C)* match, *TTP* generates NROT retrieved by *B* and NRDT retrieved by *A*; otherwise, terminates the running of the protocol, which is fair to both *A* and *B* since *B* does not get NROT and *A* does not get NRDT. At the same time, *B* does not know the content of the message either. Therefore, we can come to the conclusion that our proposed protocol is fair.

## 5. Conclusion

In this paper, we proposed a new fair non-repudiation protocol FNRP. The FNRP uses hash value of ciphertext of plaintext message instead of plaintext message itself in non-repudiation tokens, which solves the transmitting large amount of data problem in FNP and CMP and improves the confidentiality of message. In the step three and four of FNRP, we use FTP to avoid the problem of email relay in CMP. Finally, FNRP is a true fair protocol for both message sender and receiver providing the communicating channels are not disconnected forever. Non-repudiation service is providing irrefutable evidences for resolution of disputation; it can not automatically resolve disputations.

## 6. References

[1] Robert H. Deng, Li Gong, Aurel A. Lazar, and Weiguo Wang, "Practical protocols for certified electronic mail", *Journal of Network System Manager*, 4(3), 1996, pp. 279-297.

[2] J. Zhou and D. Gollmann, "A fair non-repudiation protocol", *Proceedings of 1996 IEEE Symposium on Security and Privacy*, 1996, pp. 55-61.

[3] N. Asokan, *fairness in electronic commerce*, PH.D thesis, University of Waterloo, 1998.

[4] ISO/IEC 13888.1, *Information technology - Security techniques - Non-repudiation - Part 1:Ggeneral*, 1997.

[5] ISO/IEC 13888.2, *Information technology - Security techniques - Non-repudiation - Part 2: Mechanisms using symmetric techniques*, 1997.

[6] ISO/IEC 13888.3, *Information technology - Security techniques - Non-repudiation - Part 3: Mechanisms using asymmetric techniques*, 1997.

[7] J. Zhou and D. Gollmann, "Evidence and non-repudiation", *Journal of Network and Computer Applications*, 1997.

[8] ISO/IEC DIS 10181-4, *Information technology - Open systems interconnection - Security framework in open systems, Part 4: Non-repudiation*, 1996.

[9] A. Bahreman and J. D. Tygar, "Certified electronic mail", *Proceedings of the internet society symposium on network and distributed system security*, san Diego, California, February 1994, pp. 3-19.

[10] T. Coffey and P. Saidha, "Non-repudiation with mandatory proof receipt",*Computer Communication Review*, 26(1), January 1996, pp. 6-17.

[11] R. Kailar, "Accountability in electronic commerce protocols", *IEEE transactions on software engineering*, vol. 22, No. 5, May 1996, pp. 313-328.

[12] R. Kailar, "Reasoning about accountability in protocol for electronic commerce", *Proceedings of the IEEE symposium on security and privacy*, May, 1995.