A New Method for Formalizing Optimistic Fair Exchange Protocols

Ming Chen[1], Kaigui Wu[1], Jie Xu[1,2], and Pan He[1]

[1] College of Computer, Chongqing University,
400044 Chongqing, China
[2] School of Computing, University of Leeds, UK
chenming9824@yahoo.com.cn, kaiguiwu@cqu.edu.cn,
jxu@comp.leeds.ac.uk, hopewhite68@hotmail.com

Abstract. It is difficult to analyze the timeliness of optimistic fair exchange protocols by using belief logic. For the problem, a new formal model and reasoning logic were proposed. In the new model, channel errors were attackers' behaviors, the participants were divided into honest and dishonest ones, and the attackers were attributed to two types of intruders. Based on the ideas of the model checking, the protocol was defined as an evolved logic system that has the Kripke structure. The new logic defined the time operators that describe the temporal relations among the participants' behaviors. By a typical optimistic fair exchange protocol, the article demonstrates the protocol analysis process in the new model. Two flaws were discovered and improved, which shows that the new method can be used to analyze the fairness and timeliness of optimistic fair exchange protocols.

Keywords: optimistic fair exchange, formalize analysis, logic reasoning, model checking, timeliness.

1 Introduction

The growing importance of e-commerce and the increasing number of applications in this area has lead to an increasing concern for fair exchange. Fair exchange is used to exchange valuable data fairly between two parties, and fair exchange protocols place important roles in achieving the fair exchange.

Formal analysis technology is an important means for analyzing fair exchange protocols. Early methods [1, 2] on this field mainly relied on belief logic. Belief logic can be used to prove the non-repudiability. But, it is difficult to analyze the timeliness of optimistic fair exchange protocols by using belief logic. Model checking methods solved the difficulty [3-6]. However, there is a main problem, state space explosion, in model checking methods. Although there are many methods, no way to fully describe the nature of such protocols in all. Model checking and logical reasoning technologies have their pros and cons. The current trend is to find combined methods and find out more attack strategies.

M. Soriano, S. Qing, and J. López (Eds.): ICICS 2010, LNCS 6476, pp. 251–265, 2010.

This paper aims to analyze the optimistic fair exchange protocols. The optimistic fair exchange, firstly proposed by Asokan [7], utilized an offline Trusted Third Party (TTP) to reduce its load. That is, the protocol only invokes the TTP in case of failures or conflicts. Then, studies on the optimistic fair exchange have been a hot field [7-12]. On the basis of the belief logic and model checking methods, this paper constructs a new formal method inheriting the advantages of belief logic, focuses on the model and logic definitions, and solves the communication channel and time-sensitive issues in optimistic fair exchange protocols.

The communication channel problem is important in fair exchange protocols. An exchange relies on the Internet, which is modeled as an asynchronous distributed system consisting of a set of processes. The messages in channels suffer various threats, including channel errors and active attacks of intruders who can eavesdrop, intercept, distort and delay messages. There is a widely accepted assumption that there exists three kinds' channel [7]: the reliable channel, the resilient channel and the unreliable channel. A reliable channel would deliver data correctly within a known, bounded amount of time. A resilient channel delivers data correctly after a finite, but unknown amount of time. An unreliable channel does not ensure that the message can be received correctly by the designated recipient. As a rule, in an asynchronous system, channels are resilient or unreliable. The channel between two participants is unreliable, and the channel between a participant and a TTP is resilient. Although the channel assumption simplifies the complexity of protocol design, it brings about the difficulty in protocol analysis. In [13], an unreliable channel was simulated into an attack behavior. In our article, a new formal model is advocated, and the channel problems are transferred to two Dolev-Yao [14] intruders (hereinafter abbreviated as DY intruder). The unreliable channel and the resilient channel are controlled by a standard DY intruder and a weak DY intruder, respectively.

Time-sensitive is another important issue in fair exchange protocols. Time attributes (including the order and the time point when messages arrived at the recipient) impact on the security of protocols. It is difficult to use belief logic-based formal methods, such as SVO logic [15], to analyze the protocols' timeliness, since the time attributes have not been well described in these methods. For example, Zhou and Gollmann's non-repudiation protocol [16], which had analyzed [2] by using SVO logic, had a timeliness fault pointed out by Kim et al. [17]. A time operator is introduced in the new logic to meet the needs of the timeliness analysis.

In this article, a new formal method is put forward. The method regards participants as processes that can send, receive, cache and deal with messages in an asynchronous communication environment. The method also considers running modes for fair exchange protocols as asynchronous systems that are consisted of the processes. Therefore, in our method, a fair exchange protocol is defined as an evolution system with the Kripke structure [18], and a new, BAN-like [19] logic with time operator is proposed. According to the model checking methods, we check in the system whether there exists a path which harms the security. If all existed paths maintain the fairness, the protocol is considered as correct, or else some attacks will be disclosed.

The rest parts of this paper are as follows: the second part analyzes and discusses the fair exchange's security in details; the third part describes a new formal model and a new logic; the fourth part demonstrates the deduction process for the new logic by a

case study; the fifth part reviews and summarizes the related research works; the final part sums up the research and puts forward the ideals that continue the next research.

2 Security Definition of Fair Exchange

An intuitive understanding of the security of fair exchange is given in this section, which follows the descriptions of Asokan [8] and Pagnia et al. [10]. We assume that there are two participants *A* (Alice) and *B* (Bob), and each party starts with an electronic item. There are two possible termination states of the fair exchange protocols, either a success or an abortion.

Definition 1. *If an exchange protocol respects the following three mandatory properties, we say it is secure.*

- *Effectiveness. If both parties behave honestly, when the protocol has completed, A and B have the items that they want to obtain individually, and both reach a successful termination state.*
- *Timeliness. The parties that behave according to the protocol always have the ability to reach either a successful termination state or an aborted termination state in a finite amount of time.*
- *Fairness. If at least one party does not behave according to the protocol, then, at the end of the exchange protocol run, either both parties obtain their expected items or no one can win anything valuable.*

In an asynchronous system, messages may be delayed for an arbitrary (but finite) amount of time. It is not possible, say for an honest party *A*, to distinguish the two cases [20]: the one, party *B* has sent a message but the channel has problems; the other, *B* has stopped the protocol overall. In other words, the final state of party *A* depends on an action performed by party *B*. If *B* does not perform this action, *A* will never be notified. So, owing to the asynchrony, *A* cannot decide whether a message from *B* will arrive or *B* has stopped exchanging. As long as *A* is still waiting, both outcomes (success or abortion) are possible, which is called the un-decidability of the termination state.

3 The New Formal Method

We firstly propose the intruder models of optimistic fair exchange protocols.

3.1 The Formal Model

Table 1 summarizes the behaviors of three kinds of attackers (since channel errors would harm the fairness of exchange, they are considered as the passive attackers).

According to the DY intruder model, all messages go through the intruder. The intruder can delay, modify, intercept or replay all the messages flowing in the system. A dishonest participant can also delay, forge, replay and do not send some messages to the counterparty. Channel errors give rise to delayed messages, lost messages or bit errors. As can be seen from Table 1, the three behaviors of each line cause the same

effect. Based on the channel assumption, the three attackers are attributed to two types of intruders. The one is the standard DY intruder [14] dominating all the sub-protocols without TTP; the other is the weak DY intruder dominating all sub-protocols with TTP.

Table 1. Analysis of the attackers' behaviors

	DY intruders	dishonest participants	channel errors
behaviors	delay	delay	delay(unreliable/reliable)
	modify or forge	modify or forge	bit errors (reliable)
	intercept	do not send	lose (reliable)
	replay	replay	/

Definition 2. *A weak DY intruder can delay and replay all the messages transmitted on the channels, and forge some messages that he will send to TTP. However, he cannot stop any messages to arrive at the designated recipient correctly.*

The intruder models of optimistic fair exchange protocols are categorized as three types, and shown in Fig. 1.

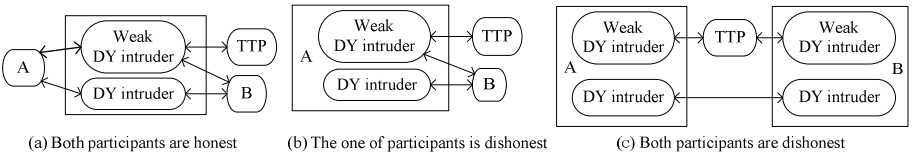

Fig. 1. The intruder models of optimistic fair exchange protocols

As can be seen in Fig. 1, we divide the participants into two parts: the honest entities and the dishonest ones. Each dishonest entity can be regarded as a (weak) DY intruder dominating the channels, and he looks forward to obtaining advantages during the exchanges by dishonest or even malicious behaviors.

Assumption 1. *If an optimistic fair exchange protocol is verifiably secure in the model (b), also in the models (a) and (c).*

In the first intruder model (shown in (a)), participants *A* and *B* are both honest, and the intruders are only external ones. In the second (shown in (b)), a dishonest party *A* is regarded as an inner intruder attacking the honest *B*. In the third model (shown in (c)), participants *A* and *B* are both dishonest, and attack each other. In models (a) and (c), the abilities of both parties are equal. In the model (b), there is a DY intruder, that is, the dishonest party *A*. *A* communicates with *B* and TTP directly, and *A* also acts as a weak DY intruder to interfere in the communication between *B* and TTP. Obviously, the model (b) is the strongest one among them, since the honest participant confronts with the greatest threat due to unequal capacities. So, we make the assumption 1.

According to Assumption1, the model (b) is adopted to analyze the fairness and the timeliness in this paper. For an honest participant, his goal is to ensure the fairness as

he always expects to receive right electronic items while paying his electronic ones. For a dishonest participant, his goal is to break the fairness by not delivering his electronic items while obtaining the desired ones.

The core idea of the new model is that each honest party determines its own status by available messages, but cannot judge behaviors and states of other parties. Thus, (weak) DY intruders are introduced to simulate this phenomenon. When the aggressive behaviors like shown in Table 1 happen, specific entities are not considered and all are reduced to behaviors of DY intruders. Then, behaviors of the dishonest participants and channel issues both turn to be behaviors of the intruder and integrate with the classical intruder model.

3.2 The New Logic

3.2.1 The FES Logic Definitions

Definition 3. *An optimistic fair exchange system (FES) is a 5-tuple* $\Sigma=\langle \Pi, \Psi, \Im, \sigma, \rho \rangle$ *where:*

- Π *is a set of participant identifiers*
- $\Im$ *is a nonempty, finite set of states, and* $\Im_0 \subseteq \Im$ *is a set of initial states*
- Ψ *is a set of formulae, and* $\Psi_{AP} \subseteq \Psi$ *is a set of atomic formulae*
- $\sigma: \Im \rightarrow 2^{\Psi}$ *is a mapping that labels each state in* $\Im$ *with the set of atomic formulae true in that state*
- $\rho \subseteq \Im \times \Im$ *is a total transition relation that is for each* $s \in \Im \setminus \exists t \in \Im$ *such that* $(s, t) \in \rho$.

FES logic is composed of the participants, states and formulae. The mappings are defined by the specific protocol. And ρ is a set transition function that maps a state and a participant to a nonempty set of choices, where each choice is a possible next state. Whenever the system is in a state s_i, each party X chooses a next state $s_j \in \rho(s_i, X)$. However, which state will be the next depends on the choices made by the other participants, because the successor of s_i must lie in the intersection of the choices made by all participants. The transition function is non-blocking, and the participants together shape a unique next state. A path in FES is a finite sequence of states $Path_{\Sigma} = s_0, s_1, s_2, \ldots, s_k$ such that $s_0 \in \Im_0$, and for all $i > 0$, $(s_{i-1}, s_i) \in \rho$.

Definition 4. *The term* $\Omega=\{\{0,1\}^{\omega}\}$ *is assumed to be freely generated from three sets:*

- $\prod \subseteq \Omega$*, which contains participant identifiers,*
- $M \subseteq \Omega$*, which contains predictable texts, and*
- $K \subseteq \Omega$*, which contains keys.*
- *The set of keys (K) is divided into four sets: encryption keys (*K_{enc}*), decryption keys (*K_{dec}*), signature keys (*K_{sig}*), and verification keys (*K_{ver}*). In addition,* K_X *says that the keys of* $X \in \prod$.

The term, being made up of the participant identifiers, keys and other messages, define the message space of fair exchange protocols. Where $\prod \cap K = \varnothing$, $\prod \cap M \neq \varnothing$ and $K \cap M \neq \varnothing$.

Definition 5. *The feasibly computable computations are as follows:*

- $H(m): \Omega \to \Omega$, *which represents hash calculation*
- $E_K(m): \Omega \times K_{enc} \to \Omega$, *which represents encryption*
- $D_K(m): \Omega \times K_{dec} \to \Omega$, *which represents decryption*
- $S_K(m): \Omega \times K_{sig} \to \Omega$, *which represents signature*
- $V_K(m): \Omega \times K_{ver} \to \Omega$, *which represents signature verification*
- $\Omega \times \Omega \to \Omega$, *which represents concatenation of terms*
- $\Omega \to \Omega \times \Omega$, *which represents splitting a concatenated term.*

Definition 6. $\Im$ *is a nonempty, finite set of state, and* $s \in \Im$ *is* $\langle A, \varepsilon, \Gamma, t \rangle$, *where:*

- $A \in \prod$ *is a participant identifier*
- $\varepsilon \in \{Ini, Wai, Act, Abo, Suc\}$ *is a description of the current process state*
- $\Gamma \subseteq \Omega$ *is a set of knowledge that A has learned at present*
- $t \in \{0,1,\ldots,n\}$ *is a discrete time element at this point.*

$\Im$, as a finite set, is made up of all the possible states in the system. The character s expresses a local state, and is labeled by a 4-tuple. Especially, $s_{A \cdot i}$ denotes the knowledge and state of the participant $A \in \prod$ at $t=i$. In this paper, the participants are defined as processes with the storage and processing powers in an asynchronous system. So the notation ε is an expression of a possible process state: initiation, waiting, activation, abortion, or success.

Definition 7. Ψ_{AP} *is a set of atomic formulae, and is made up of the followings:*

- $A \ni m$, *which represents that A generates m*
- $A \triangleright m$, *which represents that A sends m*
- $A \triangleleft m$, *which represents that A receives m*
- $A \propto m$, *which represents that A verifies m holds*
- $\oplus Timer$, *which represents that the timer starts*
- $\otimes Timer$, *says that the timer shuts down*
- $\perp Timer$, *which represents time-out*
- $A \succ \varphi$, *which represents that A believes* φ.

Where, $A \in \prod$, $m \in \Omega$, and φ is a formula. The atomic formulae define participant behaviors and time operators. The timer is introduced to trigger the process events. In the real world, the waiting time of process for the next event is limited, and it needs to be waken up in a constant time interval. So, $\perp Timer$ is used to trigger this event.

Definition 8. *A FES formula is one of the following:*

- *p, for formula* $p \in \Psi_{AP}$, *and*
- $\neg\varphi$, $\varphi \wedge \gamma$ *or* $\varphi \vee \gamma$, *where* φ *and* γ *are FES formulae.*

We now give the conditions under which a formula is assigned to be true. For a run, if the formula φ holds at a state s_t, we write '$s_t \vDash \varphi$'. It is inductively defined below. Where, *if and only if* is abbreviated to *iff*.

- $s_t \vDash A \lhd m$ *iff* m is in the set of received messages for A at s_t.
- $s_t \vDash A \rhd m$ *iff* for some message m, A sent m at $s_{t'|0 \le t' \le t}$.
- $s_t \vDash A \propto m$ *iff* A has checked m and thinks that m holds at $\forall s_{t'|t' \ge t}$.
- $s_t \vDash A \ni m$ *iff* m can be generated by A through available computations at s_t.
- $s_t \vDash \oplus Timer_A$ *iff* $s_{t'|t'=t-1} \vDash A \rhd m$, and A is going to receive some messages.
- $s_t \vDash \otimes Timer_A$ *iff* A has received rightly the messages for which he is waiting at s_t.
- $s_t \vDash \perp Timer_A$ *iff*, A has not received the messages for which he is waiting at s_t.
- $s_t \vDash A \succ \varphi$ *iff* $\exists s_{t'|0 \le t' \le t} \vDash \varphi$.
- $s_t \vDash \neg\varphi$ *iff* $s_t \nvDash \varphi$.
- $s_t \vDash \varphi \wedge \gamma$ *iff* $s_t \vDash \varphi$ and $s_t \vDash \gamma$.
- $s_t \vDash \varphi \vee \gamma$ *iff* $s_t \vDash \varphi$ or $s_t \vDash \gamma$.

This completes the conditions necessary to assign truth values to all formulae in the logic.

3.2.2 The Rules of Reasoning in FES Logic

Rules of reasoning are instances of classical propositional calculus. "$\gamma \rightarrow \varphi$" means that φ can be derived from formulae γ.

R1. $A \ni m \rightarrow m \in \Gamma_A$

R1 means A generates a new message m, and m is set into Γ_A. Here, A, according to his knowledge, can generate recursively any new message by adopting the computations defined in *Definition* 5. Especially, $\ni_s$ denotes the signature computation.

R2. $A \lhd m \rightarrow m \in \Gamma_A$

R2 means A receives a message m, and m is set into Γ_A.

R3. $(S_{k_B}(m), m, k_{B \cdot ver}) \in \Gamma_A \wedge A \propto S_{k_B}(m) \rightarrow A \succ (B \rhd m)$

R3 means if the signature $S_{k_B}(m)$, the signed message m and the verification key $k_{B \cdot ver}$ all exist in Γ_A, and A verifies the signature is correct, A believes B sent m.

3.2.3 The Semantics of FES Logic

Our logic is defined as a finite set (Ψ) of formulae, a finite set ($\Im$) of states, and a finite set ($\prod$) of participants, P_1, P_2,…, P_n. There is a participant representing the intruder. Each participant P_i has a local state s_i. A global state is thus an n-tuple of local states. Participants can perform four actions: sending a message, receiving a message, verifying a message and generating a new one, denoted by $\rhd$, $\lhd$, $\propto$ and $\ni$ respectively. While a participant sends a message, his state is transferred to waiting for receiving messages, or success (only if the exchange has been complete successfully). If the message for which a party is waiting is received correctly in a constant time interval, then his state is transferred to activation to verify the received message or to generate a new one, if not, the event, $\perp Timer$, would take place. We use $\langle event \rangle \Rightarrow \langle state - transformation \rangle$ to express these transformations, where $\langle event \rangle$ would be the actions preformed by a participant or the $\perp Timer$.

A run is an infinite sequence of global states indexed by integral times. The initial state of the current exchange is at t=0. A global state at time t in a run is indicated by $S(t)$, and is composed of all the local states at t. The local states of each participant include the state description and the sets of available knowledge. For P_i, the collection of all messages which are explicitly received, initially available and newly generated by P_i, is defined as a set of knowledge, Γ_{Pi}. Γ_{Pi} consists of the messages mentioned previously plus all the messages he can recursively generate from those messages via his available transformations. These are the computations that are feasibly computed by that party (up to the computational complexity limitations). Typically, the available transformations consist of arbitrary numbers of applications of the message formation rules that has defined in *Definition* 5. These include encryptions and decryptions with available keys as well as other functions the participant may perform, such as, hashes, signatures, etc. All participants are assumed to be able to decide the equality of any messages they can produce from what they have. For example, suppose a signature scheme is being used, such as RSA, if P_i has messages X and K (the private signature key of P_i) then P_i can generate the digital signature for X, $Y = \{X\}_K^r$.

Furthermore, we define a local timer which plays an important role in state transitions. Timer has three states: opening, closing and time-out, denoted by $\oplus Timer$, $\otimes Timer$ and $\perp Timer$ respectively. When a process who delegates a participant sends one message and waits for receiving another message, the timer opens. If the message has been received rightly by the process, the timer closes and the process state is transferred from *Wai* to *Act*. When the timer is time-out, the messages to be received are not received correctly in the scope of the pre-set maximum time, and the process will be waken up to perform the next action which is to terminate the protocol or to request the conflict resolution.

4 Case Study

This section, taking a case for example, analyzes a real world's optimistic fair exchange protocol, JAB protocol [12], by using our logic. The JAB protocol is shown in Fig. 2.

Main protocol	*Recovery protocol*
1. $O \rightarrow R : NRO, m$	1. $R \rightarrow TTP : NRR, NRO$
2. $R \rightarrow O : NRR$	2. $TTP \rightarrow R : NRA$
3. $O \rightarrow R : NRA$	3. $TTP \rightarrow O : NRA$

Fig. 2. The JAB protocol

The JAB protocol has two sub protocols, the main protocol and the recovery protocol, and has three participants. *Origin* (O) is the entity that sends the message m (e.g. the order and his credit card information) to the receiver with his digital signature, which acts as a proof of origin (Non- repudiation of origin, $NRO = S_{k_O}(m)$, where NRO is a digital signature generated by O over m). The origin also needs to perform a proof of acknowledgement (Non-repudiation of acknowledgment, $NRA_O = S_{k_O}(NRR)$), which is

an embedded signature generated over the *NRR*. *Receiver* (*R*) is the entity that receives the message *m* and the corresponding *NRO*. Once the *NRO* has been validated, the receiver applies a digital signature to the received messages. This signature is then sent to the origin as a proof of receipt (Non-repudiation of receipt, $NRR = S_{k_R}(m, NRO)$). *TTP* is a Trusted Third Party participating in the recovery protocol only when an abnormal situation occurs in the main protocol. Thus, he acts in an optimistic mode. In the last two steps of the recovery protocol, the *TTP* delivers $NRA_T = S_{k_T}(NRR)$ to both the origin and receiver. In addition, two timeouts, t_0 and t_1, play an important role in the protocol. Timeout t_0 establishes the time that the receiver must wait before executing the recovery protocol, and t_1 must be higher enough than t_0 for allowing the receiver to complete the recovery protocol. The two timeouts avoid some specific attacks on the protocol, which are further detailed by Hernandez- Ardieta et al. [12].

4.1 Formal Analysis of the JAB Protocol

Definition 9. *The JAB protocol's initial state* $s_0 = \langle X, \varepsilon_0, \Gamma_0, t_0 \rangle$ *is defined by:*

$\cdot X \in \{O, R, T, I\}, \varepsilon_{X\cdot 0} = Ini, t_0 = 0$

$\cdot \Gamma_{O\cdot 0} = \{\Pi_{O\cdot 0} = \{O, R\}, M_{O\cdot 0} = \{m, NRO\}, \mathrm{K}_{O\cdot 0} = \{k_{O\cdot sig}, k_{X\cdot ver}\}$

$\cdot \Gamma_{R\cdot 0} = \{\Pi_{R\cdot 0} = \{O, R, T\}, M_{R\cdot 0} = \varnothing, \mathrm{K}_{R\cdot 0} = \{k_{R\cdot sig}, k_{X\cdot ver}\}\}$

$\cdot \Gamma_{T\cdot 0} = \{\Pi_{T\cdot 0} = \{T\}, M_{T\cdot 0} = \varnothing, \mathrm{K}_{T\cdot 0} = \{k_{T\cdot sig}, k_{X\cdot ver}\}\}$

$\cdot \Gamma_{I\cdot 0} = \{\Pi_{I\cdot 0} = \{I\}, M_{I\cdot 0} = \varnothing, \mathrm{K}_{I\cdot 0} = \{\mathrm{K}_{X\cdot ver} \cup \{k_{I\cdot sig}\}\}\}$

The JAB protocol is a typical optimistic fair exchange protocol, and it is supposed that *O* and *R* have agreed on the contents, e.g. *m*, that will be exchanged between them. In consequence, *O* and *R* have known each other before exchange, that is, {*O*, *R*}$\subseteq\prod_{O\cdot 0}$ and {*O*, *R*}$\subseteq\prod_{R\cdot 0}$. But, $T \notin \prod_{O\cdot 0}$ holds since *O* does not know in *TTP*s who will be requested to execute the recovery protocol by *R*. In addition, $\mathrm{K}_{O\cdot 0}$, $\mathrm{K}_{R\cdot 0}$, $\mathrm{K}_{T\cdot 0}$ and $\mathrm{K}_{I\cdot 0}$are key sets that all participants has possessed at the beginning of the protocol, including their own private key used to sign and the public keys of all participants used to verify the signature.

The focus of this paper is to analyze the *Fairness* and the *Timeliness* of the JAB protocol. That is, if at least one party does not behave according to the protocol, we will check whether the *Fairness* and *Timeliness* are ensured. The check will be performed from two aspects, one is that *O* is dishonest but *R* is honest, the other is on the contrary. We firstly assume that *O* is dishonest, and conspires with *I* ($R \notin I \wedge T \notin I$) to attack the honest *R*. We notate *O* and *I* as I_O.

Theorem 1. *At the ending of the main protocol, the global state* $s_j = \langle X, \varepsilon_j, \Gamma_j, j \rangle$ *of JAB protocol is as follows:*

$\cdot X \in \{I_O, R, T\}, \varepsilon_{Io\cdot j} = Suc, \varepsilon_{R\cdot j} = Wai, \varepsilon_{T\cdot j} = Ini, \Gamma_{T\cdot j} = \Gamma_{T\cdot 0}$

$\cdot \Gamma_{Io\cdot j} = \{\Pi_{Io\cdot j} = \{I_O, R\}, M_{Io\cdot j} = \{m, NRO, NRR, NRA_O\}, \mathrm{K}_{Io\cdot j} = \{\mathrm{K}_{X\cdot ver} \cup \{k_{I\cdot sig}, k_{O\cdot sig}\}\}\}$

$\cdot \Gamma_{R\cdot j} = \{\Pi_{R\cdot j} = \{O, R, T\}, M_{R\cdot j} = \{m, NRO, NRR\}, \mathrm{K}_{R\cdot j} = \{k_{R\cdot sig}, k_{X\cdot ver}\}\}$

Proof: *It can be deduced as follows:*

$t_{Io}=1: I_O \triangleright m \,\|\, NRO \Rightarrow \varepsilon_{Io\cdot 1}=Wai \wedge \oplus Timer_{Io}$

$t_R=1: R \triangleleft m \,\|\, NRO \wedge R \propto m \wedge R \propto NRO \Rightarrow \varepsilon_{R\cdot 1}=Act \wedge (m, NRO) \in \Gamma_R$

$t_R=2: R \ni_s \wedge \triangleright NRR \Rightarrow \varepsilon_{R\cdot 2}=Wai \wedge \oplus Timer_R \wedge NRR \in \Gamma_R$

$t_{Io}=2: (\perp Timer_{Io} \Rightarrow \varepsilon^*_{Io\cdot 2}=Abo) \vee (I_O \triangleleft \wedge \propto NRR \Rightarrow$

$\quad \varepsilon_{Io\cdot 2}=Act \wedge \otimes Timer_{Io} \wedge NRR \in \Gamma_{Ic})$

$t_{Io}=3: I_O \ni_s NRA_O \Rightarrow \varepsilon_{Io\cdot 3}=Suc \wedge NRA_O \in \Gamma_{Ic}$

At this point, $\varepsilon_{Io\cdot j}=\varepsilon_{Io\cdot 3}$ and $\varepsilon_{R\cdot j}=\varepsilon_{R\cdot 2}$. T maintains the initial state because he is not involved in this stage. In addition, state $\varepsilon^*_{Io\cdot 2}$ is also possible, if there are channel errors. However, we ignore this case since I_O does not actively abandon the exchange. After I_O receives the *NRR* in step 2 of the main protocol, the final evidence NRA_O can be generated by I_O. In order to obtain an advantage, I_O must abort the main protocol, and disturb the communication between R and T.

In order to interpret the procedures of proof better, we demonstrates some course of I_O's state transformations in figure 3.

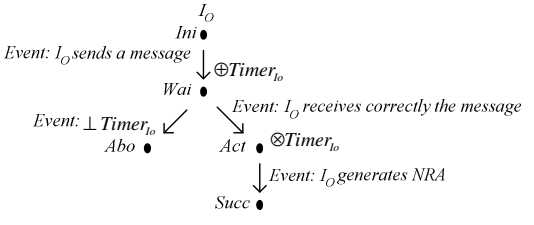

Fig. 3. An instantiation of state transformations

Theorem 2. *At the ending of the recovery protocol, the global state* $s_l=\langle X, \varepsilon_l, \Gamma_l, l\rangle$ *of JAB protocol is as follows:*

$\cdot X \in \{I_O, R, T\}, \varepsilon_{Io\cdot l}=Suc, \varepsilon_{R\cdot l}=Suc, \varepsilon_{T\cdot l}=Suc$

$\cdot \Gamma_{Io\cdot l}=\{\Pi_{Io\cdot l}=\{I_O, R\}, M_{Io\cdot l}=\{m, NRO, NRR, NRA_T\}, \mathrm{K}_{Io\cdot l}=\{\mathrm{K}_{X\cdot ver} \bigcup \{k_{I\cdot sig}, k_{O\cdot sig}\}\}\}$

$\cdot \Gamma_{R\cdot l}=\{\Pi_{R\cdot l}=\{O, R, T\}, M_{R\cdot l}=\{m, NRO, NRR, NRA_T\}, \mathrm{K}_{R\cdot l}=\{k_{R\cdot sig}, k_{X\cdot ver}\}\}$

$\cdot \Gamma_{T\cdot l}=\{\Pi_{T\cdot l}=\{O, R, T\}, M_{T\cdot l}=\{NRO, NRR, NRA_T\}, \mathrm{K}_{T\cdot l}=\{k_{T\cdot sig}, k_{X\cdot ver}\}\}$

Proof: *It can be deduced as follows:*

$t_R=3: \perp Timer_R \wedge R \triangleright NRR \wedge \neg(R \triangleleft NRA_O) \Rightarrow \varepsilon_{R\cdot 3}=Act$

$t_R=4: R \triangleright NRR \,\|\, NRO \Rightarrow \varepsilon_{R\cdot 4}=Wai \wedge \oplus Timer_R$

$t_T=1: T \triangleleft NRR \,\|\, NRO \wedge T \propto NRR \wedge T \propto NRO \Rightarrow \varepsilon_{T\cdot 1}=Act \wedge (NRR, NRO) \in \Gamma_T$

$t_T=2: T \ni_s \wedge \triangleright NRA_T \Rightarrow NRA_T \in \Gamma_T \wedge \varepsilon_{T\cdot 2}=Suc$

$t_R=5: R \triangleleft \wedge \propto NRA_T \Rightarrow \varepsilon_{R\cdot 5}=Suc \wedge NRA_T \in \Gamma_R$

$t_{Io}=4: I_O \triangleleft \wedge \propto NRA_T \Rightarrow \varepsilon_{Io\cdot 4}=Suc \wedge NRA_T \in \Gamma_{Io}$

In the above reasoning, the state transition, from $\varepsilon_{R,2}$ to $\varepsilon_{R,3}$, is carried out by R to executing the recovery protocol at t_R=3. In terms of the protocol, if R is honest, he must wait at least t_0 when the *NRR* has been sent at the step 2 in the main protocol.

Theorem 1 and *Theorem* 2 indicate that the JAB protocol is secure in our model on the basis of the assumption that there is an intruder conspired with the dishonest party O, since we cannot find a path to disclose that the dishonest participant O can win the goal that he breaks the *Fairness* of JAB protocol.

On the other hand, we assume that R is dishonest, and conspires with I ($O\notin I \wedge T\notin I$) to attack the protocol. R and I are notated as I_R.

Theorem 3. *At the ending of the JAB protocol, the global state is* $s_i = \langle X, \varepsilon_i, \Gamma_i, i\rangle$ *as follows:*

$\cdot X \in \{O, I_R, T\}, \varepsilon_{O\cdot i} = Abo, \varepsilon_{Io\cdot i} = Suc, \varepsilon_{T\cdot i} = Suc$

$\cdot \Gamma_{O\cdot i} = \{\Pi_{O\cdot i} = \{O, R\}, M_{O\cdot i} = \{m, NRO\}, \mathrm{K}_{O\cdot i} = \{k_{O\cdot sig}, k_{X\cdot ver}\}$

$\cdot \Gamma_{Ir\cdot i} = \{\Pi_{Ir\cdot i} = \{O, I_R, T\}, M_{Ir\cdot i} = \{m, NRO, NRR, NRA_T^*\}, \mathrm{K}_{Ir\cdot i} = \{\mathrm{K}_{X\cdot ver} \bigcup \{k_{I\cdot sig}, k_{R\cdot sig}\}\}\}$

$\cdot \Gamma_{T\cdot i} = \{\Pi_{T\cdot i} = \{O, R^*, T\}, M_{T\cdot i} = \{NRO, NRR, NRA_T^*\}, \mathrm{K}_{T\cdot i} = \{k_{T\cdot sig}, k_{X\cdot ver}\}\}.$

Proof: *It can be deduced as follows:*

$t_O = 1 : O \rhd m \parallel NRO \Rightarrow \varepsilon_{O\cdot 1} = Wai \wedge \oplus Timer_O$

$t_{Ir} = 1 : I_R \lhd m \parallel NRO \wedge I_R \propto NRO \Rightarrow \varepsilon_{Ir\cdot 1} = Act \wedge NRO \in \Gamma_{Ir}$

For the purpose of having an advantage, I_R aborts the main protocol, and executes the recovery protocol.

$t_{Ir} = 2 : I_R \ni_s NRR_{Ir} \wedge I_R \rhd NRR_{Ir} \parallel NRO \Rightarrow \varepsilon_{Ir\cdot 2} = Wai \wedge \oplus Timer_{Ir} \wedge NRR_{Ir} \in \Gamma_{Ir}$

$t_T = 1 : T \lhd NRR_{Ir} \parallel NRO \wedge T \propto NRR_{Ir} \wedge T \propto NRO \Rightarrow$

$\quad \varepsilon_{T\cdot 1} = Act \wedge (NRR_{Ir}, NRO) \in \Gamma_T$

$t_T = 2 : T \ni_s \wedge \rhd NRA_T \Rightarrow NRA_T \in \Gamma_T \wedge \varepsilon_{T\cdot 2} = Suc$

$t_{Ir} = 3 : I_R \lhd \wedge \propto NRA_T \Rightarrow NRA_T \in \Gamma_{Ir} \wedge \varepsilon_{Ir\cdot 3} = Suc$

$t_O = 2 : (\perp Timer_O \Rightarrow \varepsilon_{O\cdot 2} = Abo) \vee (O \lhd \wedge \propto NRA_T \Rightarrow NRA_T \in \Gamma_O \wedge \varepsilon_{O\cdot 2}^* = Suc)$

In the above reasoning, the states $\varepsilon_{O\cdot 2}^*$ and $\varepsilon_{O\cdot 2}$ are both reachable, which results in the un-decidability of the termination state that has been described in *Section* 2. If the waiting time of O is not long enough at t_O=2, and the weak DV intruder delays the time when the term NRA_T reaches O, then $s|t_O{=}2 \vDash (\varepsilon_{O\cdot 2}{=}Abo \wedge NRA_T \notin \Gamma_O)$. In this case, *Theorem* 3 is true.

Theorem 3 shows that the JAB protocol does not achieve the *Timeliness* and *Fairness*, and the attack strategy is shown in Fig. 4.

So, another timeout, t_2, must be defined to establish the time when O has to wait after sending the message *NRO*. Moreover, t_2 must be higher than t_1, and the *TTP* has to record and publish the termination states of all runs distinguished by their identifier, which allow O to obtain the termination state by means of accessing *TTP*. In the attack scene shown in Fig. 4, O can know that NRA_T has been sent out by *TTP*. Then, he will wait continually until NRA_T is received by him or ask the *TTP* to send it again. It is shown in Fig. 5.

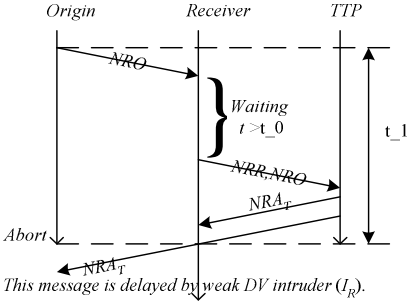

Fig. 4. An attack on JAB protocol

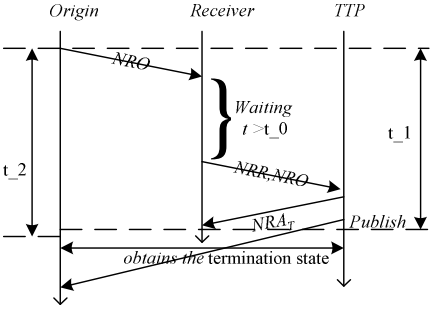

Fig. 5. The improvement of JAB protocol

There is another defect exposed by the *Theorem* 3 that NRR_{Ir} can be generated arbitrarily with the key known by I_R. The reason for forming this fault is that, from the received messages (*NRO* and *NRR*), *T* cannot decide which one is the legitimate party consulted with *O*. We redefine the $NRO = S_{K_O}(m \| O \| R)$, which remedies it.

5 Related Works

Formal technology is an important means to analyze the fair exchange protocols. Based on the BAN logic [19], a new logic was put forward by Kailar [1]. The Kailar's logic can be used to prove non-repudiability, but not fairness of protocols. Zhou and Gollmann's work [2] using the SVO logic only studied non-repudiability, but neither fairness nor timeliness. Other approaches, such as the one by Schneider [21] using CSP or by Bella and Paulson [22] using inductive methods, both studied non-repudiability as well as fairness and timeliness. However, they did not study optimistic protocols. Protocols using an in-line or on-line TTP seem to be much easier to analyze, as the protocols are not branching.

More intensive studies were realized by Shmatikov and Mitchell [3] using the finite state model-checker Murφ, by Kremer and Raskin [4] using the game-theoretic approaches based on the ATL (Alternating-time Temporal Logic), by Cederquist and

Torabi Dashti [5] using the process algebraic intruder model, and by Basagiannis et al. [6] using the SPIN model-checker based on the intrusion attack tactics.

5.1 Analysis and Comparison

SVO logic [15] is a significant belief-logic-based method. Analyzing the fairness and the timeliness of optimistic fair exchange protocols by adopting SVO logic is, nevertheless, uneasy. There are two main reasons. Firstly, there exist different intruder models. SVO logic employs the classical DY model and does not distinguish the resilient channel with the unreliable channel. SVO logic always assumes that both parties are honest and does not consider the internal attack behavior of dishonest entities. Not only is the new intruder model of optimistic protocols defined, but also channel issues are transferred and behavior attributions of participants are distinguished in our article. Secondly, it is hard to describe the fairness of optimistic protocols by SVO logic. For optimistic protocols have not only one branch and one available terminal result, the goal is hardly described brief by SVO logic. Hereby, protocols in this article are defined as the system that will evolve by time. The model checking method can verify the whole running status of any protocol and avoid presetting the goal for a run.

Shmatikov and Mitchell [3] used the model-checker Murφ to analyze the ASW [8] and the GJM [9] protocols. The protocol participants as well as the intruder were described by using the Murφ input language by hand. Similar to ours, they modeled a dishonest participant by asking him to collaborate with a classical DY intruder. They also modeled different kinds of channel. Unlike we did, they used the invariants to express properties conditional on the message delivery sent to resilient channels. Cederquist and Torabi Dashti [5] formalized resilient channels, and presented a process-algebraic model of intruder as well as the pattern of properties for checking fairness. Furthermore, their intruder model had been implemented by using a synchronous model, and been used to verify the fairness of non-repudiation protocols. The synchronous model simplified the intruder description and the intruder implementation, but did not describe asynchronous network enough.

Compared to the above methods, the method used in this article has following characteristics: first, the participants' behavior attributes are distinguished in the formal models; second, the weak DV intruder is introduced to describe resilient channels; third, our scheme combines the belief logic with the model checking method to successfully analyze the fairness and timeliness of optimistic fair exchange protocols. But, the automatic inference technology still needs to be studied. A key issue for achieving automatic model-checker is to automatically generate the intrusion messages, such as the $NRR_{I'}$ (in *Section* 4.1). Our ideal is that two finite-state automata called as Message Generator and Message Inspector respectively will be implemented. The former is used to produce messages. Its input is the knowledge of requestor, and its output is a set of terms T ($T \subseteq \Omega$) that is computed recursively via all available computations has been defined in *Definition* 5. The latter is used to inspect whether a term, which is received by the requestor or is a new one produced by Message Generator, is valid at this point in the current run of protocol. Its input is a term, and output is a bit $b \in \{0,1\}$ that denotes true or false.

6 Conclusion

In this article, we have shown that exchange protocols, due to the behaviors of their participants, are modeled as three kinds. The fairness and the timeliness of optimistic fair exchange protocols should be analyzed in the model (b) defined in 3.1. In this model, the channel problems are mapped to two types of DY intruders. Then, a new logic is proposed as well. Case study shows that the new logic characterizes a variety of party acts, and describes the constraints and temporal relationships among different acts. Meanwhile, two attacks are revealed, which indicates that the JAB protocol does not have the strong fairness as their authors alleged. The improvements of these attacks are also put forward.

Though the research on a formal method of fair exchange protocols has been proceeding soundly, its validity remains to be further validated by formalizing others exchange protocols. In addition, the effectiveness analysis of the exchange protocols seems to be easier, and can be performed in the model (a) defined in 3.1 by using our logic. Our studies will continue along this way.

Acknowledgment. We would like to thank anonymous referees for their useful comments. This work is supported in part by the Major Research plan of the National Natural Science Foundation of China under Grant No.90818028.

References

1. Kailar, R.: Accountability in electronic commerce protocols. IEEE Transactions on Software Engineering 5, 313–328 (1996)
2. Zhou, J., Gollmann, D.: Towards verification of non-repudiation protocols. In: International Refinement Workshop and Formal Methods Pacific, Canberra, Australia, pp. 370–380. Springer, Heidelberg (1998)
3. Shmatikov, V., Mitchell, J.C.: Finite-state analysis of two contract signing protocols. Theoretical Computer Science 2, 419–450 (2002)
4. Kremer, S., Raskin, J.: A game-based verification of non-repudiation and fair exchange protocols. Journal of Computer Security 3, 399–429 (2003)
5. Cederquist, J., Torabi Dashti, M.: An intruder model for verifying termination in security protocols. Technical Report 05-29, CTIT, University of Twente, Enschede, The Netherlands (2005)
6. Basagiannis, S., Katsaros, P., Pombortsis, A.: Intrusion Attack Tactics for the model checking of e-commerce security guarantees. In: Saglietti, F., Oster, N. (eds.) SAFECOMP 2007. LNCS, vol. 4680, pp. 238–251. Springer, Heidelberg (2007)
7. Asokan, N.: Fairness in Electronic Commerce. PhD thesis, University of Waterloo (1998)
8. Asokan, N., Shoup, V., Waidner, M.: Asynchronous protocols for optimistic fair exchange. In: Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, CA, pp. 86–99. IEEE Computer Society Press, Los Alamitos (1998)
9. Garay, J.A., Jakobsson, M., MacKenzie, P.: Abuse-free optimistic contract signing. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 449–466. Springer, Heidelberg (1999)
10. Pagnia, H., Vogt, H., Gärtner, F.C.: Fair Exchange. The Computer Journal 1, 55–76 (2003)

11. Wang, G.: Generic non-repudiation protocols supporting transparent off-line TTP. Journal of Computer Security 5, 441–467 (2006)
12. Hernandez-Ardieta, J.L., Gonzalez-Tablas, A.I., Alvarez, B.R.: An optimistic fair exchange protocol based on signature policies. Computers & Security 10, 309–322 (2008)
13. Qing, S., Li, G.: A formal model of fair exchange protocols. Science in China Ser. F Information Sciences 4, 499–512 (2005)
14. Dolev, D., Yao, A.C.: On the security of public key protocols. IEEE Transactions on Information Theory 2, 198–208 (1983)
15. Syverson, P.F., Van Oorschot, P.C.: An unified cryptographic protocol logic. NRL Publication 5540-227, Naval Research Lab, Washington, DC, USA (1996)
16. Zhou, J., Gollmann, D.A.: Fair non-repudiation protocol. In: Proc. of the 1996 IEEE Symp. on Security and Privacy, Oakland, CA, pp. 55–61 (1996)
17. Kim, K., Park, S., Baek, J.: Improving fairness and privacy of Zhou-Gollmann's fair non-repudiation protocol. In: Proc. of the 1999 ICPP Workshop on Security (IWSEC), Aizu, Japan, pp. 140–145 (1999)
18. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press, Cambridge (1999)
19. Burrows, M., Abadi, M., Needham, R.: A Logic of Authentication. ACM Transactions in Computer systems 1, 18–36 (1990)
20. Fischer, M.J., Lynch, N.A., Paterson, M.S.: Impossibility of distributed consensus with one faulty process. Journal of the ACM, 374–382 (1985)
21. Schneider, S.A.: Formal analysis of a non-repudiation protocol. In: 11th IEEE Computer Security Foundations Workshop, Washington- Brussels-Tokyo, pp. 54–65. IEEE, Los Alamitos (1998)
22. Bella, G., Paulson, L.C.: Accountability protocols: Formalized and verified. ACM Trans. Inf. Syst. Secur. 2, 138–161 (2006)