Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Fairness analysis of extra-gain guilty of a non-repudiation protocol

Xu GUO

Department of Electronics and Information, Shanghai Dianji University, Shanghai 201306, China

E-mail: guox@sdju.edu.cn

Received Aug. 29, 2021; Revision accepted Dec. 17, 2021; Crosschecked Mar. 23, 2022

Abstract: Many traditional applications can be refined thanks to the development of blockchain technology. One of these services is non-repudiation, in which participants in a communication process cannot deny their involvement. Due to the vulnerabilities of the non-repudiation protocols, one of the parties involved in the communication can often avoid non-repudiation rules and obtain the expected information to the detriment of the interests of the other party, resulting in adverse effects. This paper studies the fairness guarantee quantitatively through probabilistic model checking. E-fairness is measured by modeling the protocol in probabilistic timed automata and verifying the appropriate property specified in the probabilistic computation tree logic. Furthermore, our analysis proposes insight for choosing suitable values for different parameters associated with the protocol so that a certain degree of fairness can be obtained. Therefore, the reverse question—for a certain degree of fairness ε, how can the protocol parameters be specified to ensure fairness—is answered.

Key words: Non-repudiation; Fairness analysis; Probabilistic model checking; PRISM
https://doi.org/10.1631/FITEE.2100413 **CLC number:** TP301.2

1 Introduction

1.1 Background

With the rapid growth of smart terminals, blockchain technology has been widely studied (Dinh et al., 2018). This technology not only provides an acentric infrastructure in different fields, including medicine (Griggs et al., 2018; Christodoulou et al., 2020; Resiere et al., 2020), Internet of Things (Novo, 2018; Hang and Kim, 2019), economics (Esfahani and Mohammed, 2018; Allen et al., 2020), software engineering (Litchfield and Herbert, 2018; Erhan et al., 2019; Mendiboure et al., 2020), and even Covid-19 fighting (Yu et al., 2021), but also maintains continuous and tamper-resistant data records in chronological order. Moreover, it has become a hot issue in both academia and industry (Wu et al., 2018; Deng et al., 2019; Feng et al., 2019). With the wide application of blockchain, fairness requirements have appeared in almost all the application scenarios, such as the issues of double payment and hard fork, which are caused by block conflicts (Pérez-Solà et al., 2019), and directly affect the integrity and effectiveness of E-commerce transactions.

Repudiation is defined as the denial by one party of having been involved in the whole or part in a communication process (Fig. 1). For the originator, which is a service provider, repudiation means that it denies having sent a message. For a recipient that acts as a client, repudiation refers to its denial of having received the message. Fairness requires that two parties involved in the communication be treated equally, and that neither of the parties can obtain an advantage over the other. One party can reach a state where it can terminate the communication process to maintain fairness without relying on the actions of the other. To solve the problem, the

ORCID: Xu GUO, https://orcid.org/0000-0003-0803-8620

protocols that can offer non-repudiation services are called non-repudiation protocols. Non-repudiation is one of the vital security services in blockchain technology. It is applied mainly in electronic commerce and message transmission systems (Ruland and Sassmannshausen, 2015).

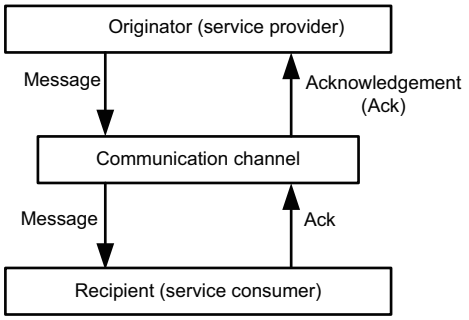

Fig. 1 Communication model

This paper studies the fairness of the non-repudiation protocol proposed by Markowitch and Roggeman (1999). It offers non-repudiation services without the involvement of a trusted third party (TTP). The fairness property of this protocol is ensured by a given degree of fairness, tolerance ε. That is, either both parties receive their expected items, or the probability that a malicious party obtains any valuable information, while the other party achieves nothing, is less than ε, where $\varepsilon \in [0, 1]$. The protocol can guarantee non-repudiation with certain probabilities, where tolerance ε is affected by different protocol parameter values. Because different configurations may produce different fairness problems, this paper uses probabilistic model checking to analyze the fairness quantitatively. Specifically, this paper studies the following questions: (1) Does parameter configuration affect the fairness? If so, what is the impact? (2) If the recipient behaves dishonestly, can the fairness be guaranteed? (3) How should the parameters be set to ensure fairness? The participants in the non-repudiation protocol are modeled as two separate units composed of the originator (O) and the recipient (R) in the setting of probabilistic model checking. They are described in the specification of PRISM, which is a probabilistic model checker.

1.2 Related works

The related works summarized in this study fall into four categories:

1. Design of non-repudiation protocols

The ownership of a radio frequency identification (RFID) tags item changes frequently during its lifetime, but few protocols address the non-repudiation problem. Piramuthu (2017) proposed a non-repudiation protocol for transferring ownership of tagged items. Because blockchain technology can provide an unchangeable data registration system, the services must provide fair, certified notifications that require a fair exchange of values. The data are composed of a message and the proof of non-repudiation origin. Mut-Puigserver et al. (2018) provided two solutions that allow certified notations to be sent when it is necessary to keep secret or register the contents of the notice.

2. Security analysis of non-repudiation protocols

Chatterjee and Raman (2014) studied the automatic synthesis of a fair non-repudiation protocol by applying game-theoretic controller synthesis. In addition to model checking, assume-guarantee strategy can discover the vulnerabilities of a designed protocol using counter-examples in synthesis. The purpose of Chatterjee and Raman (2014)'s work was to use non-repudiation as an example to show that uncertainty methods based on information flow theory are not enough to verify the security of cryptographic protocols.

3. Security assurance of non-repudiation protocols

Sarr et al. (2019) proposed an identification scheme based on a combination of the Rivest–Shamir–Adleman (RSA) algorithm and a strongly unforgeable signature scheme. The scheme is non-interactive, non-repudiation and is shown to be inside secure under the RSA assumption and the random oracle model. Roscoe and Ryan (2017) studied how to approximate fair exchange without a TTP for password authenticated key exchange (PAKE) protocols. They considered two scenarios: in the first scenario, two parties exchanged stochastically and in the second scenario, there was an intruder that can guess the password with probability ε.

4. Fairness and privacy protection strategies in the context of non-repudiation

Decentralization is a mainstream approach for privacy protection and fair exchange. Li et al. (2021) presented a fair big data exchange scheme, which can guarantee that buyers and sellers can complete transactions fairly and autonomously without a TTP. To enhance security and privacy, Li et al. (2021) also

applied an m-out-of-n oblivious transfer protocol on the transactions, and applied an Ether cheque system to guarantee fairness and autonomy. Baza et al. (2021) proposed B-Ride, a decentralized ride-sharing service, based on a public blockchain. The aim of B-Ride is to ensure (1) the tradeoff between trip information sharing and personal privacy protection, (2) the accountability under anonymity, and (3) fair payment without a TTP.

The above studies can greatly enhance non-repudiation protocol security, check security problems in the protocol, and balance the tradeoff between privacy protection and fair data sharing, but do not provide quantitative fairness analysis of non-repudiation protocols. The aim of this work is to fill in this gap.

1.3 Contributions

The contributions of this paper can be summarized as follows:

1. Fairness analysis is very important for blockchain technology, but there is little literature on this aspect. To fill in the gap, this paper provides a quantitative analysis of the fairness of Markowitch and Roggeman (1999)'s non-repudiation protocol.

2. Three evolutionary versions of the protocol are considered. That is, both parties' compliance with the protocol and the recipient's deception are considered. Moreover, the recipient's ability to deceive is enhanced with improved processing ability.

3. The protocol's fairness performance under different parameter configurations is presented and relevant opinions on the protocol design are put forward according to the results of quantitative analysis.

2 Preliminaries

2.1 Related concepts

This subsection focuses on the background techniques to be used in formal verification of the non-repudiation protocol that follows.

1. Fairness

Fairness is a very important factor because it has nothing to do with network size or complexity. The goal of fairness is to guarantee that benefits are distributed fairly among the participants in an activity (Zhao et al., 2016; Américo et al., 2018; Said and Cristescu, 2020). Informally, fairness means that at each step of the protocol, either both parties receive their expected items, or neither of them receives any valuable information about their expected items.

2. Distribution

For a finite set S, a discrete probability distribution over S is a function $f: S \to \{0,1\}$ such that $\Sigma_{s\in S}\ f(s)=1$. $\mathrm{Dist}(S)$ is the set of all distributions over finite subsets of S. The point distribution f_m denotes the distribution which assigns probability 1 to m. The support of f denoted by $\mathrm{supp}(f)$ is the set $\mathrm{supp}(f) \triangleq \{s \in S | f(s) > 0\}$.

3. Evaluation

V is a certain set of variables, and each element $v \in V$ has a specific domain Dom. The domain of the variables is restricted to integers, Boolean or bounded integers. A valuation of V is a function val: $V \to \cup_{v\in V}\mathrm{Dom}$, where $\mathrm{val}(v) \in \mathrm{Dom}$. The set of expressions over set V is denoted as Expr_v. For an expression $e \in \mathrm{Expr}_v$ and a valuation $v \in V$, the valuation of the expression is denoted by $\|e\|_v$, where each occurrence of variable v is substituted by the value $\mathrm{val}(v)$ and the term is evaluated afterward. For a Boolean expression, $\|\mathrm{ber}\| \in \{0,1\}$, where "0" for false and "1" for true. Given a variable x and a Boolean expression ber, denote $x \models \mathrm{ber}$ if and only if $\|\mathrm{ber}\|_x = 1$.

4. Clock restrictions

Clock restrictions are a special kind of variables that can model real-time behaviors. Let C be a finite set of variables named clock which take values from $\mathbb{R}^+$ (non-negative reals). A clock valuation is referred to as a point $v \in R^C$. For a clock variable $c \in C$, $v(c)$ stands for the value of v assigned to c. For any $v \in R^C$ and $u \in \mathbb{R}^+$, the clock valuation defined as $v(c)+u$ is denoted as $u+v$, which may represent the elapsed time. For $B \subseteq C$, let $v[B:=0]$ denote the clock valuation gained from v by resetting all clocks to 0. Given a finite set of clock variables V, a clock restriction τ is defined inductively by the following syntax:

$$\begin{aligned}\tau ::= &\ \mathrm{false}|\mathrm{true}|v \le e|v\\ &= e|v \ge e|u+e \le v+f|\neg\tau|\tau\wedge\tau,\end{aligned}$$

where $u, v \in V$ and $e, f \in \mathbb{N}$. The set of all clock restrictions over V is denoted as $\mathrm{CR}(V)$. The clock valuation v satisfies the restriction $\mathrm{CR}(V)$, written as $v \propto \mathrm{CR}(V)$, if and only if $\mathrm{CR}(V)$ resolves to true

after each clock value $v(c)$ from v takes the place of the corresponding clock $c \in C$.

2.2 Probabilistic timed automaton (PTA)

Definition 1 (Probabilistic timed automaton, PTA) A PTA is defined as a tuple (L, C, inv, PR, Lab) described as follows:

(1) L is a finite set of locations;

(2) C is a finite set of clock variables;

(3) inv: $L \to \mathrm{CR}(C)$ is the invariant condition;

(4) $\mathrm{PR} \subseteq L \times \mathrm{CR}(C) \times \mathrm{Dist}(2^C \times L)$ is a finite set of probabilistic edge relations;

(5) Lab: $L \to 2^{\mathrm{AP}}$ is a labeling function that assigns atomic propositions to locations, where AP is a finite set of action labels.

A state of a PTA is defined as a pair $(l, c) \in L \times R^C$, where $c \propto \mathrm{inv}(l)$.

2.3 Probabilistic timed computation tree logic (PTCTL)

The properties of a PTA can be specified by the probabilistic timed computation tree logic (PTCTL). Following timed computation tree logic (TCTL) (Henzinger et al., 1994), PTCTL uses a set of formula clocks Z, which are separate from clocks C of the PTA. Formula clocks are assigned values by a formula clock valuation $\delta \in R^Z$. Timing restrictions are expressed using such clocks and the reset operator is $z.\phi$. Similar to probabilistic computation tree logic (PCTL) (Hansson and Jonsson, 1994), PTCTL contains the probabilistic quantifier $P_{\sim\lambda}[\cdot]$.

Definition 2 (PTCTL) The syntax of PTCTL is defined as follows: $\phi ::= \mathrm{true} \mid a \mid \zeta \mid z.\phi \mid \phi \wedge \phi \mid \neg\phi \mid P_{\sim p}\,[\phi \cup \phi]$, where $a \in \mathrm{AP}$ is an atomic proposition, $\zeta \in \mathrm{CR}(C \cup Z)$, $z \in Z$, "$\sim$" $\in \{<, \leq, >, \geq\}$, and $p \in [0, 1]$.

3 Non-repudiation protocol

This section briefly describes the non-repudiation protocol proposed by Markowitch and Roggeman (1999). As a kind of fair exchange protocol, a non-repudiation protocol guarantees that when the signature exchange over a network is complete, neither party can deny its participation in the process. If the protocol terminates successfully, it can provide each participant with evidence of commitment that cannot be denied by the other party. As illustrated in Fig. 1, this protocol can guarantee a fair exchange between the service provider, the originator O, the service consumer, and the recipient R. Assume that an authentication process, which is useful in resolving possible disputes, is executed before the protocol.

The protocol starts when R sends a request to O asking for a message. To prevent reply attacks, R chooses date D as time stamp along with the request. The message to be sent by O is divided into n fragments according to a geometric distribution with parameter p. The integer n stands for the number of rounds it takes to send the message; i.e., one fragment is transmitted in each round. In the remainder of this paper, the transmitted message fragment is called "message." For the sake of fairness, n will never be revealed to the recipient during the protocol execution. O computes n functions $f_1, f_2, \cdots, f_n$, which are parts of a function composition. They can help split the requested message M:

$$f_n(M_n) \bowtie f_{n-1}(M_{n-1}) \bowtie \cdots \bowtie f_1(M_1) = M.$$

The constitution operator "$\bowtie$" does not satisfy the commutative law to ensure that R cannot obtain the requested message until the last message $f_1(M_1)$ has been received. At step i, on receiving a request, O sends the encrypted message M_{n-i+1} to R and waits for the corresponding acknowledgement from R. To ensure fairness and prevent R from decoding the message, if O does not receive an acknowledgement within a certain period, it will terminate the protocol and declare that R is suspected of cheating. The chosen time limit AD must be greater than the time it takes for R to return an acknowledgement, but it ought to be less than the time required for R to decode the message. The messages encrypted by a participant P using a private key are denoted as $Ey_P(M)$. The encryption of message M under key K is denoted as $\{M\}_K$. The execution process of this protocol can be described as follows:

(1) R selects date D as the time stamp st;

(2) $R \to O$: Ey_R(request, R, O, st), the originator checks st then;

(3) $O \to R$: $Ey_O(\{M_1\}_K, O, R, \mathrm{st})$;

(4) $R \to O$: $Ey_R(\mathrm{ack}_1)$;

(5) $1-p$: $O \to R$: $Ey_O(\{M_i\}_K, O, R, \mathrm{st})$;

(6) $R \to O$: $Ey_R(\mathrm{ack}_i)$;

(7) if $i<n$, then goto step (4);

(8) p: $O \to R$: $Ey_O(\{M_n\}_K, O, R, \mathrm{st})$;

(9) $R \to O$: $Ey_R(\text{ack}_n)$.

The notion "$R \to O$: msg" means that a message msg is sent by R and received by O. At step (4), O makes a probabilistic choice to guarantee $p = \varepsilon$, where ε is a parameter used to maintain the fairness. O randomly sends R a piece of message with probability $1 - p$, where p is a parameter of a geometric distribution. After receiving an acknowledgement message from R, O continues to send messages until the last one. At step (7), O sends the last message encrypted by K with probability p. On receiving the last acknowledgement, O terminates the protocol correctly. One can assume that each acknowledgement may contain the following content "R confirms having received message M_i." This can be implemented easily because each acknowledgement may contain a hash of message M_i.

According to the above description, the protocol should achieve the following informal non-repudiation requirements:

1. Fairness

The originator's non-repudiation is pledged by the messages it sent which are signed with the private key. The recipient's non-repudiation is represented by the last acknowledgement, ack_n. If the protocol can terminate after O receives ack_n, both parties can obtain the expected information and the protocol is fair. If the protocol terminates because one party achieves the expected information while the other gains nothing, the protocol is unfair. If the protocol terminates after O receives ack_i $(i \neq n)$, then neither O nor R obtains the expected items and fairness is maintained. In brief, the protocol should guarantee that during execution, each party can obtain the expected information, or the probability that a malicious party can obtain the expected items while the other gains nothing should be no more than ε.

2. Time-limited

The network transmission delay can be ignored. The participants can always reach a point, within a certain period with probability 1, where they can terminate the protocol for the sake of fairness.

3. Determinacy

If both parties obey the protocol and behave correctly, they can both obtain the expected items when the protocol terminates with probability 1. Neither party involved in the protocol can achieve its own goal while preventing the other party from satisfying its goal.

4. Tasks

The implementation of a non-repudiation service includes evidence generation, signature transmission and storage, authentication, dispute resolution, and a duration for each transaction.

5. Efficiency

This refers to the TTP's degree of participation. The TTP may be non-existent, off-line, or on-line. In the on-line mode, the TTP fully participates in the execution of the protocol. In this mode, the TTP acts as a message delivery agent; all messages are passed through the TTP, thus creating a potential bottleneck, as proposed in the Zhou–Gollmann protocol (Zhou and Gollman, 1996). As an optimistic approach to fair exchange, proposed by Kremer and Markowitch (2000) and Kremer et al. (2002), in off-line mode, the TTP simply monitors the execution of the protocol and appears only when there is repudiation to be resolved. As an improved scheme, non-repudiation protocols without TTP have been proposed (Markowitch and Roggeman, 1999; Cederquist et al., 2005; Piramuthu, 2017).

There are other details about the original protocol that have no direct impact on modeling. Due to space constraints, they are omitted.

4 Modeling

In this section, according to the requirements of probabilistic model checking, the non-repudiation protocol described in Section 3 is modeled to verify the properties of the protocol.

4.1 Discussion

Both parties involved in the non-repudiation protocol exhibit probabilistic and non-deterministic behaviors that are time-related. Hence, the protocol should be modeled as PTA. To this end, we can use the following assumptions:

(1) Encryption scheme: Because the non-repudiation properties to be verified depend on the number and order of messages that are exchanged, the results will be extracted from the encryption schemes used by the original protocol and the message exchanging process is simply modeled between the two involved parties.

(2) Transmission delay: Although the original protocol does not require communication channel quality of service (QoS), the transmission delay

should also be considered. If a packet is assumed to be delayed by a participant, it will not arrive at the other party's location. One variable is used to represent the time cost of sending a message.

(3) TTP: The original protocol does not require a TTP. In this situation, the originator and the recipient are in an ad hoc pattern. The two involved parties should be modeled independently and have self-contained clocks.

(4) Shared variables: The involved parties have their own clocks, but some actions should be obeyed globally. To achieve this goal, the model should provide communal variables that represent the corresponding clock restraints, which can be accessed by the involved parties.

(5) The number of transmission rounds: The originator randomly chooses a number r, which is the number of rounds it will take to send the message. According to the original protocol, r obeys a geometric distribution of parameter p. This may lead to infinite steps, so for practical purposes, assume that the originator can set a maximum number of rounds $r_{\max}$. The actual number of rounds should be $\min\{r, r_{\max}\}$.

(6) Parametrization: The formal model encapsulates as many behaviors as possible. One of the advantages of the model checking technique is that it can simulate the behaviors of the target system under actual scenarios by building a flexible model. Hence, the model should be flexible by parametrization. The parameters required by the model are summarized in Table 1.

Table 1 Parameter list

Parameter	Description
p	The probability that a message is the last one
q	The probability that R can decode a message
p_{loss}	Message loss probability
ad	Time cost that R sends an ack
AD	Deadline that O can wait for an ack
DECODE	Time cost for simple malicious R to decode a message
DC_1	Time cost for powerful malicious R to decode a message
DC_2	Time cost for R's probabilistic decoding

4.2 Honest recipient version

This subsection provides the model for O and an honest recipient R. R strictly follows the protocol. The originator, as shown in Fig. 2, is ready to send messages after initialization. Upon receiving a request, O begins to send a message encrypted with key K and waits for the acknowledgement sent by R. Then, at state O_2, with probability $1-p$, it sends another encrypted message, sets the local clock x to zero, and reaches state O_1, and with probability p, it sends the last message containing K and reaches state O_3. Because the model does not describe the value that is passed, the actions are simply called "msg." At state O_2, the expiration of the deadline for receiving an acknowledgement from the recipient is modeled by the action "stop" when the local clock x affirms a value greater than AD. The protocol terminates fairly when O receives the acknowledgement for the last message. An unfair termination of the protocol is reached if and only if O does not receive the acknowledgement within AD time units. In such a case, O executes the action "stop." The constraint AD is used to represent an estimate of the maximum transmission delay of an acknowledgement. Actions such as "init" in the model are treated as instant actions.

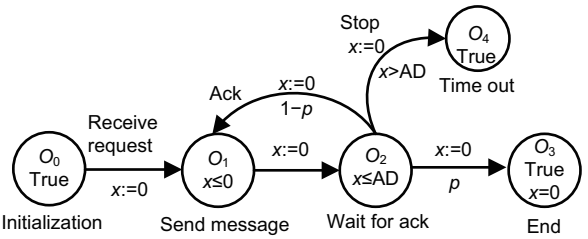

Fig. 2 An originator model

Fig. 3 shows the honest recipient model. R is a passive participant. It starts the protocol by sending a request. After receiving a message, it reaches state R_2 and begins to send an acknowledgement. It resets the local clock y to zero and goes back to state R_1 to receive another message. Whenever it receives a message, it returns an acknowledgement. Because R behaves honestly, it can always send an acknowledgement back within AD. The time t that an acknowledgement requires to be sent back and a new message to be transmitted via the network may hold: $\text{ad} \leq t < \text{AD}$.

The whole model which represents the protocol is defined as $O||R$, where O and R synchronize through the actions listed in the set A={req, msg, ack}. The protocol ends in a correct fair pattern

when O sends out the last message and receives the corresponding acknowledgemen; that is, the system reaches the state $\{O_3, R_1\}$. On one hand, if both parties behave correctly, the unfair case cannot appear. On the other hand, it is possible to find a malicious recipient who has received the expected information and denied that fact.

4.3 Malicious recipient version

This subsection considers two versions of the protocol where the recipient behaves maliciously. One is for a simple malicious recipient (SMR) R, where the recipient tries to guess which is the last message. R sends an acknowledgement and decodes the received messages at the same time. However, the success condition is limited by whether O sends the last message; that is, the probability of successful decoding is p. The decoding process needs to be completed within DECODE time units where DECODE<AD so that O will not terminate the protocol or report R's malicious behaviors. Fig. 4 shows the model for R. The time necessary to decode a message is within the interval (0, DECODE]. Due to its limited ability, R can decode the whole message if and only if it receives the last message with probability p, i.e., as shown in Fig. 4, the transition from state R_3 to state R_4. State R_4 represents the state at which R correctly decodes the whole message. Note that if DECODE<AD, R has enough time to decipher the message. Moreover, it makes no sense for O to terminate the protocol because it did not receive the last acknowledgement when the recipient has already obtained the whole message. If R fails to decode the last message with probability $1-p$, it reaches state R_5 and sends an acknowledgement to O. If O has k packages to send, the probability of unfair action is $P(x=k)=p(1-p)^{k-1}$.

Another version involves a more powerful R with a probabilistic encoder that can decode the message with probability q before O times out. For this version, R sends an acknowledgement and decodes the received message at the same time. Because R's capability is enhanced, it does not need to wait for the last message to arrive; it can successfully decode a message in DC_2 with probability q. Or, after receiving the last message, it can decode the message correctly in DC_1 with probability p. Fig. 5 shows the model for the more powerful malicious recipient. The time necessary for R to decode the message probabilistically is DC_2, which is less than AD. It takes DC_1 time units for the more powerful R to decode the message successfully. For the sake of fairness, the protocol should choose a suitable AD so that AD< DC_1. This can guarantee that R does not have enough time to decode the message during the execution of the protocol. To summarize, R can decode the message in DC_1 with probability p; it can also decode the message in DC_2 with probability q, where $DC_2 < DC_1$.

5 Fairness analysis

This section discusses, in detail, the fairness analysis of the non-repudiation protocol quantitatively. It provides the model of the protocol described in the PRISM language, discusses the properties described in the PCTL, and resolves the results of an experiment conducted with PRISM. The

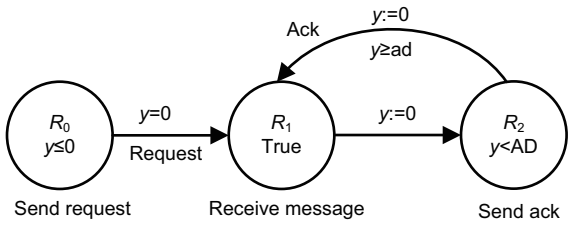

Fig. 3 An honest recipient model

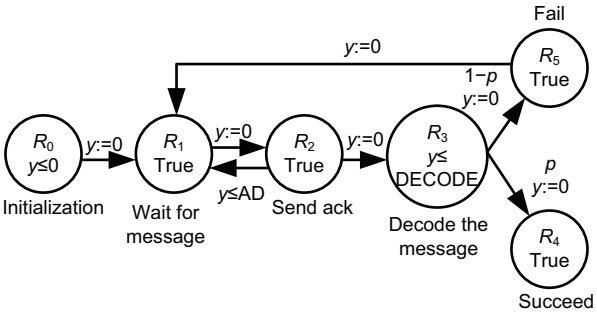

Fig. 4 A simple malicious recipient (SMR) model

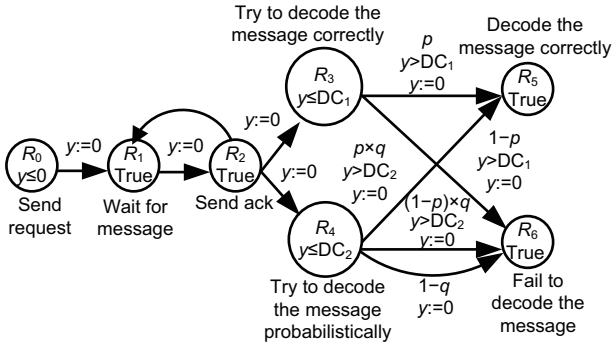

Fig. 5 A more powerful malicious recipient model

originator randomly selects the number of rounds according to the geometric distribution with parameter p. This may lead to an infinite loop. As a liveness property, termination indicates that the protocol session terminates as expected. This section first verifies the termination property and then checks the fairness if the protocol sessions can terminate.

5.1 Honest recipient

The behaviors of the originator O and the honest recipient R are modeled in Figs. 2 and 3, respectively. The corresponding PRISM codes for the models are as follows:

```
module originator
  o: [0..4];
  x: clock;
  invariant
    (o=0 => true)  & (o=1 => x<=0)
    & (o=2 => x<=AD)
    & (o=3 => true) & (o=4 => true)
  endinvariant
  [req] o=0 -> (o'=1) & (x'=0);
  [message] o=1 & x<=0 -> (o'=2);
  [ack] o=2 & x<AD ->1−p: (o'=1) & (x'=0)+p: (o'=3) & (x'=0);
  [ ] o=2 & x>=AD -> (o'=4) & (x'=0);
endmodule
module recipient
  r: [0..2];
  y: clock;
  invariant
    (r=0 => y<=0) & (r=1 => true)
   & (r=2 => y<AD)
  endinvariant
  [req] r=0 & y=0 -> (r'=1);
  [message] r=1 -> (r'=2) & (y'=0);
  [ack] r=2 & y>=ad -> (r'=1) & (y'=0);
endmodule
```

As illustrated in Fig. 2, the originator model has five states, represented by O_i $(i = 0, 1, \cdots, 4)$. Similarly, for the recipient, variable R_i $(i = 0, 1, 2)$ indicates its state. Variables x and y are local clock variables for O and R, respectively.

The invariant structure describes the clock invariants for each module in the PRISM model in terms of an expression. The restrictions imposed on the clock variables regulate the acceptable values, which rely on the other non-clock variable values. The clock invariants are usually used to describe each PTA location separately. The clock references must take the form of conjunctions of simple clock restrictions. That is, the simple expressions are of the form $x \sim c$ or $x \sim y$, where x and y are clock variables, c is an integer-valued expression, and "$\sim$" $\in \{<, \leq, >, \geq, =\}$. In the model shown in Fig. 2, staying at location O_1, the local clock x must satisfy the condition $x \leq 0$.

For the above participants, the honest recipient is a passive one, and it executes only a simple action loop of receiving a message and sending an acknowledgement back. The originator is in a dominant position; it can decide whether to terminate the protocol or not. When the protocol ends in a fair correct way, the originator arrives at location O_3. An honest recipient is in a completely passive position during the communication process. The termination of the protocol depends entirely on the originator, but the PTA model checking currently supports only $P_{\min=?}$ and $P_{\max=?}$ properties. The termination properties are described as follows:

$$P_{\min=?}[F\ o = 3], \quad P_{\max=?}[F\ o = 3]. \tag{1}$$

When the originator sends the last message with probability p, the verification results of the termination probability are gathered in Table 2. The experimental results are calculated by PRISM without a rounding operation. The execution and termination are determined entirely by the originator. It can terminate the protocol when it receives the last acknowledgement. Based on the above analysis, the protocol can be terminated with probability 1 when there is no deception on the part of the recipient. For each value of p, it takes the model checker 0.05 s to calculate the maximum and minimum probabilities. According to the experimental results, if the participants act according to the protocol strictly via a non-lossy channel, the protocol can terminate with probability 1.

Table 2 Termination probabilities

p	$p_{\max}$	$p_{\min}$
0.01	1.0	1.0
0.02	1.0	1.0
0.05	1.0	1.0
0.10	1.0	1.0
0.20	1.0	1.0
0.50	1.0	1.0
0.70	1.0	1.0
0.90	1.0	1.0

As for a lossy channel, assume that the messages transmitted by the channel may be lost with probability p_{loss}. What impact will the channel QoS have on the protocol termination? The termination probabilities in three cases, where p takes the values of 0.1, 0.5, and 0.9 and p_{loss} varies from 0.01 to 0.10, are shown in Fig. 6.

The termination probability of a lossy channel is affected by two factors: p and p_{loss}. When the originator sends the last message with a lower probability p, the communication time will be prolonged compared with that in the case with a high probability p, because the protocol will not terminate until the originator receives the last acknowledgement. The actual situation caused by the decrease in p is that more packets are sent. When p_{loss} is certain and more packets are sent, more packets are lost. In this way, for the same p_{loss}, the decrease in p leads to a decrease in the termination probability. Similarly, for the same p, with an increase in p_{loss}, the termination probability decreases. That is, the lower the value of p, the longer the communication time and the lower the termination probability.

As can be concluded from the previous experiments, the protocol can terminate with probability 1 in an ideal case, but the duration of the termination changes with the change of probability p. As a further step, it is necessary to consider whether the protocol can terminate in a bounded period. The property that the protocol can successfully terminate within T time units can be described in PCTL specification as follows:

$$\begin{aligned} &P_{\min=?}[F \leq T\ o = 3], \\ &P_{\max=?}[F \leq T\ o = 3]. \end{aligned} \tag{2}$$

The trends of the minimum and maximum termination probabilities are plotted in Fig. 7, where p takes the values of 0.01, 0.05, and 0.10. Other experimental results about the minimum and maximum probabilities are listed in Table 3. Obviously, regardless of the maximum or minimum value, the termination probability increases with an increase in the protocol execution time. If the execution time is long enough, the difference between the maximum and minimum probabilities becomes smaller and smaller

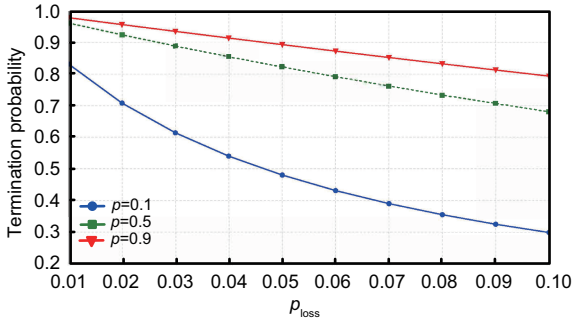

Fig. 6 Termination probability for a lossy channel

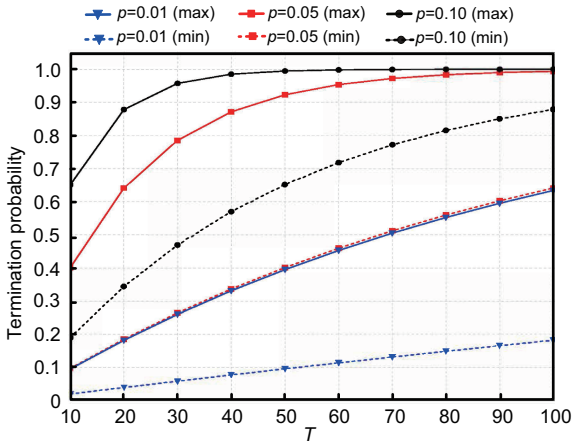

Fig. 7 Termination probability within T time units

Table 3 The minimum and maximum probabilities for other values of p

p	Minimum probability									
	$T=10$	20	30	40	50	60	70	80	90	100
0.02	0.040	0.078	0.114	0.149	0.183	0.215	0.246	0.276	0.305	0.332
0.07	0.135	0.252	0.353	0.440	0.516	0.581	0.638	0.687	0.729	0.766
0.20	0.360	0.590	0.738	0.832	0.893	0.931	0.956	0.972	0.982	0.988
0.40	0.640	0.870	0.953	0.983	0.994	0.998	0.999	1.000	1.000	1.000
0.60	0.840	0.974	0.996	0.999	1.000	1.000	1.000	1.000	1.000	1.000
p	Maximum probability									
	$T=10$	20	30	40	50	60	70	80	90	100
0.02	0.183	0.332	0.455	0.554	0.636	0.702	0.757	0.801	0.838	0.867
0.07	0.516	0.766	0.887	0.945	0.973	0.987	0.994	0.997	0.999	0.999
0.20	0.893	0.988	0.999	1.000	1.000	1.000	1.000	1.000	1.000	1.000
0.40	0.994	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
0.60	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

until it disappears completely. This is further confirmed by the experimental results. Moreover, the maximum and minimum termination probabilities increase with the increment of p. The same conclusion is confirmed by the results in Table 3. An efficient way to reduce execution time is to increase the value of p. If the originator splits the origin message into n pieces, the probability p that a message is the last one is $p = 1/n$. That is, to increase the value of p, the value of n should be reasonable.

For the lossy channel, as shown in Fig. 6, the termination probability is affected by p and p_{loss}. To study the influence of these two parameters on termination probability, it is assumed that parameter p is constant and that p_{loss} changes, and vice versa. The minimum and maximum termination probabilities in T time units are shown in Fig. 8, where p_{loss}=0.02, 0.06, and 0.10 and $p = 0.25$. The minimum termination probabilities in other circumstances are listed in Table 4. As the results show, with the increment of time, the termination probability slightly increases. If $p_{\text{loss}} = 0.10$, the termination probability cannot even exceed 0.3 within 100 time units when $p = 0.10$. As shown in Fig. 8, the termination probability, either the minimum or the maximum, increases as p_{loss} decreases. When p_{loss} is lower than 0.05, the increment is more obvious. This shows that the protocol's running time can be reduced by improving QoS of the communication channel. Moreover, as the existing experimental results show, both the maximum and the minimum probabilities increase rapidly within 40 time units, and then tend to be stable. One can therefore consider that the median execution time of the protocol is 40 time units.

The situation where p changes while p_{loss} is constant is also considered, and the experimental results are shown in Fig. 9, where p_{loss}=0.01 and p=0.01, 0.03, 0.05, and 0.10. When p_{loss} is invariant, the termination probability increases with the increase in p. In other words, for a certain period, improving the termination probability can be achieved by increasing the value of p. This is consistent with the previous experimental conclusion of improving the termination probability for a non-lossy channel. In summary, to improve the termination probability in a limited execution period, one can increase the value of probability p or improve QoS of the communication channel.

5.2 Simple malicious recipient (SMR)

In addition to the termination property, a more important issue is to discover the probability that will allow the recipient to cheat the originator in different situations, or conversely, to determine the best originator strategy to maintain fairness. This subsection begins to consider fairness analysis for

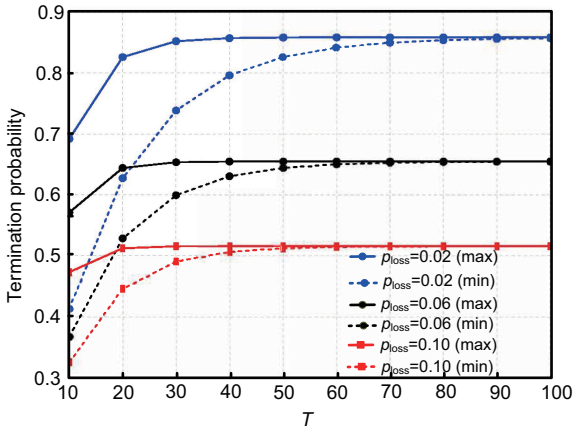

Fig. 8 Termination probability within T time units for lossy channels ($p = 0.25$)

Table 4 Minimum termination probability for other values of p and p_{loss}

p	p_{loss}	Minimum termination probability									
		$T = 10$	20	30	40	50	60	70	80	90	100
0.01	0.01	0.019	0.037	0.055	0.071	0.086	0.100	0.114	0.126	0.138	0.149
	0.05	0.017	0.031	0.042	0.050	0.057	0.063	0.067	0.071	0.074	0.076
	0.09	0.015	0.025	0.032	0.037	0.040	0.042	0.043	0.044	0.045	0.045
0.05	0.01	0.095	0.177	0.248	0.309	0.363	0.409	0.449	0.484	0.514	0.541
	0.05	0.084	0.145	0.191	0.224	0.248	0.266	0.280	0.289	0.297	0.302
	0.09	0.074	0.120	0.148	0.166	0.176	0.183	0.187	0.190	0.192	0.193
0.10	0.01	0.184	0.328	0.440	0.527	0.594	0.647	0.688	0.720	0.744	0.764
	0.05	0.164	0.271	0.343	0.390	0.421	0.441	0.455	0.463	0.469	0.473
	0.10	0.140	0.214	0.254	0.275	0.286	0.292	0.295	0.297	0.298	0.298

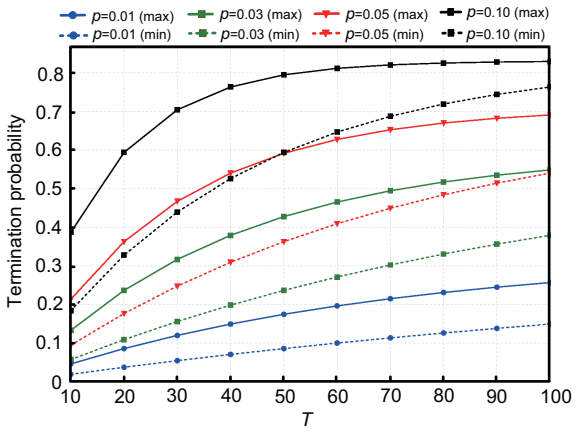

Fig. 9 Termination probability within T time units for lossy channels (p_{loss}=0.01)

two versions of the protocol, where the recipient acts maliciously. In the first version, an SMR tries to decode the message on receiving it and sends an acknowledgement within AD time units. The behavior of the SMR is shown in Fig. 4 and the PRISM codes are as follows:

```
module malicious_recipient
  invariant
     (r=0 => y<=0) & (r=1 => true) & (r=2 => true)
     & (r=3 => y<=DECODE)
     & (r=4 => true) & (r=5 => true)
  endinvariant
  [req] r=0 & y=0 -> (r'=1);
  [message] r=1 -> (r'=2) & (y'=0);
  [ack] r=2 -> (r'=1):
  [ ] r=2 -> (r'=3) & (y' =0);
  [ ] r=3 & y>=DECODE -> p: (r'=4) & (y'=0)
                +(1−p): (r'=5) & (y'=0);
  [ack] r=5 -> (r'=1) & (y' =0);
endmodule
```

As illustrated in Fig. 4, SMR has six states. Variable R_i $(i = 0, 1, \cdots, 5)$ stands for the SMR state. Variable y serves as the SMR's local clock. At state R_2, the recipient tries to decode the message on receiving it and sends an acknowledgement to the originator within AD time units. Due to the limitation of the protocol, it is possible for the recipient to decode the message successfully only when it receives the last message. That is, the probability that it can decode the message is p, where DECODE $= 8 >$ AD $= 5$.

The first property is to verify the maximum probability that the recipient can decode the message successfully. This means that in the SMR model, as shown in Fig. 4, there exists a path, starting from state R_0, by which the PTA can reach state R_4 eventually. The property can be described in PCTL as

$$P_{\max=?}[F\ r = 4]. \tag{3}$$

The recipient can decode the message only when it receives the last message with probability p, and the experimental results prove this fact, as shown in Fig. 10. The experimental results also demonstrate that the protocol is ε-fair, where $\varepsilon = p$. To maintain ε-fairness, the originator should allocate the number of packets reasonably and keep the number of packets secret. In essence, the recipient can decode the message and gain information by guessing whether the message is the last one. As can be expected, after calculation, the minimum probability is zero. So, only the maximum probability is discussed.

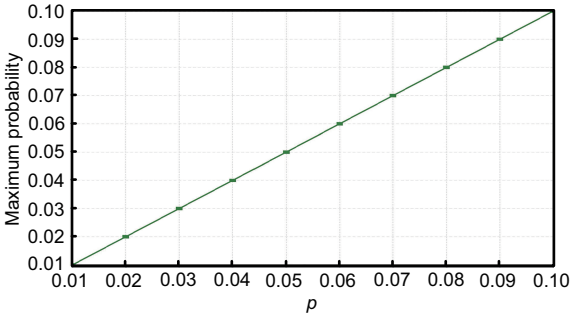

Fig. 10 Maximum probability

In practical applications, the actual situations are not always optimistic for the recipient. The recipient benefits from the fact that it can decode the message within a certain period, so the property is to verify the probability that the malicious recipient can decode the message within T time units. It can be described in the form of PCTL as follows:

$$P_{\max=?}[F \leq T\ r = 4]. \tag{4}$$

For this version, whether the recipient can successfully decode the message totally depends on whether it has received the last message. The action of the recipient is just like extracting a specific fragment from n fragments. This can be viewed as non-replacement sampling of finite samples in probability theory, which obeys hypergeometric distribution. Assume that there are n fragments to be sent; $n-1$ of them are normal fragments, and one is the last fragment. The probability that the last fragment is selected in the i^{th} round is $1/n$. As shown in Fig. 11, the maximum probability of decoding a

message remains constant after the arrival of the first message, because each message may be the last one and nothing can be achieved by waiting for a long time.

In the above experiments, it takes the recipient DECODE (>AD) time units to decode the message. It seems that the probability of success totally depends on probability p. One may assume that the recipient is more powerful, which means that less time is required to decode the message. In the following experiment, DECODE is set to 4, which is less than AD. That is, the recipient has enough time to decode the message before sending an acknowledgement. Thus, fairness will be greatly damaged. It is necessary to verify the impact on the fairness of the protocol when the value of DECODE becomes smaller, where it reduces from 8 to 4. The experiment compares the maximum probabilities that the receiver can successfully decipher a message in DECODE time units when $p = 0.01$ and 0.10. Fig. 12 shows the experimental results.

DECODE is reduced from 8 to 4, while AD remains at 5, which means that the recipient has more time to decode a message. That is, the probability

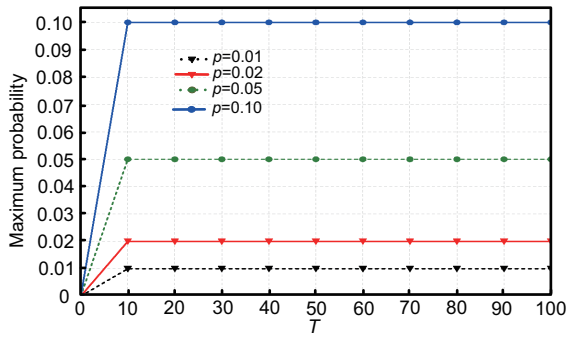

Fig. 11 Maximum probability that the message can be deciphered within T time units

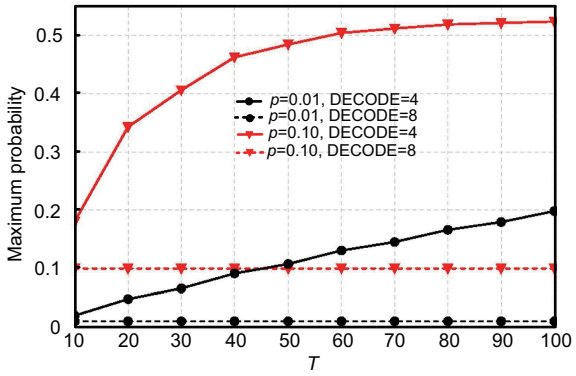

Fig. 12 Maximum probability that the recipient can decipher the message in DECODE time units

that the recipient can decode the message and obtain information increases sharply. This gives us an important insight; that is, to maintain fairness, one should fully consider the value of AD, to not only leave enough time for the recipient to send acknowledgements, but also prevent the recipient from being idle for a long time.

5.3 More powerful malicious recipient

This subsection considers a more powerful malicious recipient, which can invoke a method with probability q that it can decode the message within DC_2 ($DC_2 < AD$) time units. Moreover, it can decode the message correctly within DC_1 time units ($DC_1 > AD$). The behaviors of this kind of recipient are shown in Fig. 5 and the formal description in PRISM specification is as follows:

```
module malicious_recipient
  invariant
     (r=0 => y<=0) & (r=1 => true) & (r=2 => true)
     & (r=3 => y<=DECODE)
     & (r=4 => true) & (r=5 => true)
  endinvariant
  [req] r=0 & y=0 -> (r'=1);
  [message] r=1 -> (r'=2) & (y'=0);
  [ack] r=2 -> (r'=1):
  [ ] r=2 -> (r'=3) & (y' =0);
  [ ] r=3 & y>=DECODE -> p: (r'=4) & (y'=0)
                +(1−p): (r'=5) & (y'=0);
  [ack] r=5 -> (r'=1) & (y' =0);
endmodule
```

Because the recipient has become more powerful, the first property is to verify the maximum probability that the recipient can decode the message. This can be described in terms of PCTL as follows:

$$P_{\max=?}[F\ r = 5]. \tag{5}$$

Fig. 13 shows the trends of the maximum probability that the recipient can decipher the message. The variation of q from 0 to 1.0 represents the fact that the decoding ability of the recipient gradually improves. The probabilities for other cases are listed in Table 5. As the experimental results show, even if the recipient can decode the message within DC_2 time units, the probability is still subjected to parameter p.

With the incremental ability of the malicious recipient, can fairness be guaranteed? The probability that the recipient can decode the message within T time units can answer this question. To maintain consistency with the previous experiments (Fig. 13),

the values of p and q remain. Fig. 14 shows the maximum probability that the recipient can decode the message within T time units.

As shown in Fig. 14, the growth of the maximum probability is generally rapid within the interval [10, 20], while the growth tends to be steady in the interval [20, 100]. The time it takes to decipher the message probabilistically is DC_2, which is far less than 20. Even after a longer period, the probability is still steady. Hence, one can draw a conclusion that if the recipient decodes a message successfully, the median time cost is 20 time units. According to the analysis of the experimental results in Fig. 13 and Table 5, the probability of successful decoding increases with the increase of p and/or q. It is necessary to study the trend for the first 10 time units. The maximum probability where $T \in [1, 10]$ is shown in Fig. 15. The maximum probability that the recipient can decode the message within T time units for other cases is listed in Table 6.

Based on the above experimental results shown in Fig. 14 and Table 6, one can conclude that when $T \in [DC_2, DC_1)$, the maximum probability is approximately equal to $p \times q$. To guarantee fairness, the most direct means is to reduce the probability.

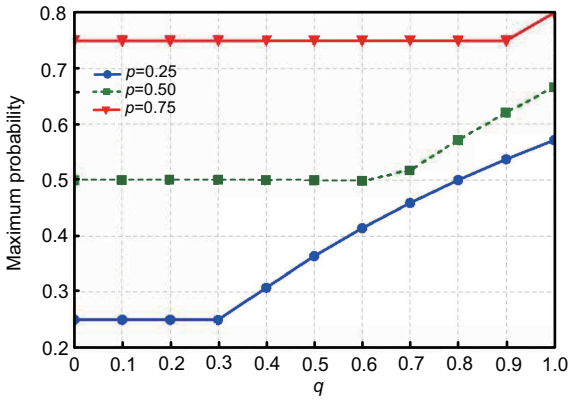

Fig. 13 Maximum probability that the recipient can decode the message as q varies

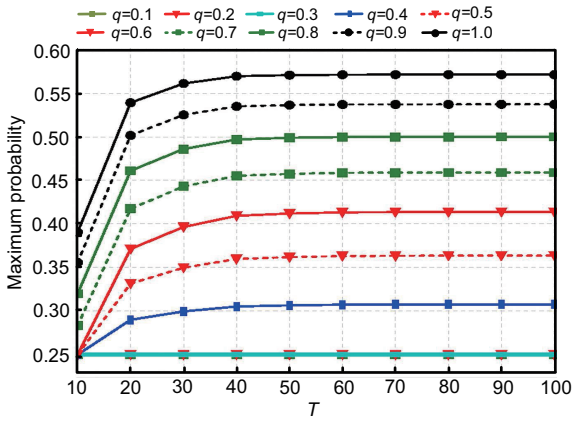

Fig. 14 Trends of the maximum probability where $T \in [10, 100]$

Table 5 Maximum probability for other certain cases

p	Maximum probability									
	q=0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.01	0.091	0.167	0.231	0.287	0.334	0.376	0.413	0.446	0.476	0.503
0.02	0.091	0.167	0.232	0.287	0.336	0.378	0.415	0.448	0.478	0.505
0.05	0.091	0.168	0.233	0.290	0.339	0.382	0.420	0.455	0.485	0.513
0.10	0.100	0.169	0.236	0.294	0.345	0.390	0.429	0.465	0.497	0.526
0.30	0.300	0.300	0.300	0.312	0.370	0.423	0.470	0.513	0.552	0.588
0.70	0.700	0.700	0.700	0.700	0.700	0.700	0.700	0.700	0.709	0.769

Table 6 Maximum termination probability for other cases of p

p	T	q=0.1	q=0.2	q=0.3	q=0.4	q=0.5	q=0.6	q=0.7	q=0.8	q=0.9	q=1.0
0.01	4	0.001	0.002	0.003	0.004	0.005	0.006	0.007	0.008	0.009	0.010
	7	0.001	0.002	0.003	0.004	0.005	0.006	0.007	0.008	0.009	0.010
	10	0.010	0.010	0.010	0.010	0.010	0.012	0.014	0.016	0.018	0.020
0.05	4	0.005	0.010	0.015	0.020	0.025	0.030	0.035	0.040	0.045	0.050
	7	0.005	0.010	0.015	0.020	0.025	0.030	0.035	0.040	0.045	0.050
	10	0.050	0.050	0.050	0.050	0.050	0.058	0.067	0.076	0.086	0.095
0.10	4	0.010	0.020	0.030	0.040	0.050	0.060	0.070	0.080	0.090	0.100
	7	0.010	0.020	0.030	0.040	0.050	0.060	0.070	0.080	0.090	0.100
	10	0.100	0.100	0.100	0.100	0.100	0.111	0.129	0.146	0.164	0.181

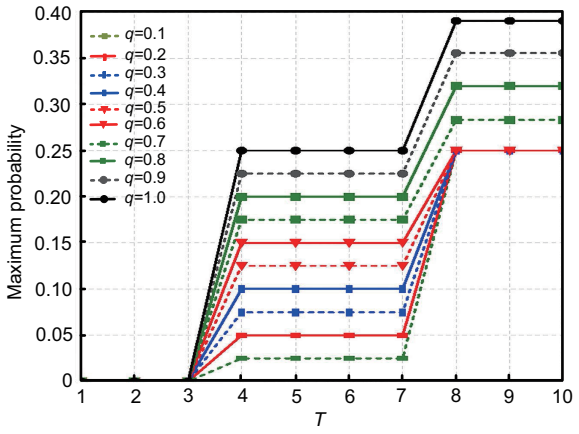

Fig. 15 Trends of the maximum probability where $T \in [1, 10]$

When the capacity of the receiver cannot be limited, the value of p can be reduced. Another way to guarantee fairness is to reduce the value of AD. If AD is far samller than DC_2, it is almost impossible for the recipient to decode the message. Another important issue to be noted is that, as listed in Table 6, the maximum probability varies sharply when T changes from 7 to 10, no matter what value q takes. This fact shows that if AD is within 7, the probability that the recipient can decipher the message is reduced significantly.

5.4 Other issues

The parameter p used in the original protocol is the parameter of a geometric distribution. Some research, such as Aldini and Gorrieri (2002), assumed that the originator can split the transmitted message following a Bernoulli distribution with parameter p. I do not think this is appropriate, because a Bernoulli distribution is a discrete probability distribution. It describes a random event that happens only once and may randomly yield two results. While the originator in the protocol is constantly sending messages, Bernoulli distribution describes only whether the message being sent is the last one or not, but cannot describe the sending behavior continuously. Moreover, the behavior cannot be described by the n-fold Bernoulli distribution, that is, a binomial experiment, because if the originator sends the last message with probability p, it means the termination of the sending process. However, the n-fold Bernoulli experiment allows the event to continue to occur with probability $1-p$, which is obviously inconsistent with the actual behavior of the originator. The behavior of the originator is exactly consistent with the behavior described by the geometric distribution; that is, the first $k-1$ experiments fail (sending normal messages), and the k^{th} (the last) experiment succeeds (sending the last message).

6 Conclusions

The non-repudiation protocol proposed by Markowitch and Roggeman (1999) provides a security solution for fair message exchange mechanisms. Related research focuses mainly on the design of non-repudiation protocols, security analysis, and security guarantees. This paper focuses on a fairness analysis of this protocol. In the context of non-repudiation protocols, fairness means that during the execution of the protocol, either the participants obtain their excepted items, or the probability that one of the parties can obtain its expected items while the other gains nothing is smaller than a negligible threshold ε.

The non-repudiation protocol is probabilistic. Therefore, this paper uses probabilistic model checking to verify the fairness properties quantitatively using the PRISM model checker. I consider three versions of this protocol. The originator is considered honest and will always abide by the protocol. In the first version, both parties obey the protocol strictly. The second version involves a simple malicious recipient that guesses whether it has received the last message and tries to decipher the message. The last version involves a more powerful recipient that can decode the message probabilistically within the deadline for sending an acknowledgement, but also deciphers the message correctly within a certain period. The three versions are modeled as PTAs and translated into PRISM specifications.

In the first version, both parties abide by the protocol strictly and fairness can be guaranteed. Because the original protocol specifies that the parameter p obeys a geometric distribution, this may lead to an infinite loop. Therefore, this paper proposes to limit the number of transmitted messages in a reasonable range and focuses on verification of the termination property. Termination over a period is affected mainly by two factors: one is the probability p, and the other is the QoS of the communication channel. Experimental results show that the higher the value of p, the higher the termination probability

within T time units, and the better the QoS, the higher the termination probability.

For the malicious recipient version, when the recipient can decode the message by only guessing whether it is the last message, the probability of unfairness is determined by parameter p. We can further restrict the recipient's idle time by setting the value of AD to ensure the fairness. The more powerful recipient can both decipher the message probabilistically within DC_2 with probability q and decode the message correctly in DC_1. The maximum probability that the recipient can decode the message is approximately equal to $p \times q$ during the period $T \in [DC_2, DC_1)$. When the computing power of the recipient cannot be limited and the probability p is not changed, we can maintain fairness by setting the value of AD so that it cannot exceed 7 and is far smaller than DC_2.

With the increase of computing power, both the originator and the recipient may infer the possible behaviors of the other party based on historical data and make corresponding decisions accordingly. Because both the originator and the recipient have the ability to think, modeling for both parties and analyzing relevant attributes are possible future work.

To prevent an infinite execution loop, this paper assumes that the upper bound of message transmission cannot exceed $r_{\max}$. This is a safety hazard. If the maximum number of steps is known or deduced by the recipient, it knows the information when the maximum number of steps is reached during an execution of the protocol. Therefore, it can refuse to acknowledge receipt of the encrypted message and break fairness with probability 1. In the future work a better scheme will be discussed for the originator to determine the number of steps in the protocol.

Compliance with ethics guidelines

Xu GUO declares that he has no conflict of interest.

References

Aldini A, Gorrieri R, 2002. Security analysis of a probabilistic non-repudiation protocol. Proc 2nd Joint Int Workshop on Process Algebra and Probabilistic Methods, Performance Modeling and Verification, p.17-36. https://doi.org/10.1007/3-540-45605-8_3

Allen DWE, Berg C, Markey-Towler B, et al., 2020. Blockchain and the evolution of institutional technologies: implications for innovation policy. *Res Policy*, 49(1):103865. https://doi.org/10.1016/j.respol.2019.103865

Américo A, Alvim MS, McIver A, 2018. An algebraic approach for reasoning about information flow. Proc 22nd Int Symp on Formal Methods, p.55-72. https://doi.org/10.1007/978-3-319-95582-7_4

Baza M, Lasla N, Mahmoud MMEA, et al., 2021. B-Ride: ride sharing with privacy-preservation, trust and fair payment atop public blockchain. *IEEE Trans Netw Sci Eng*, 8(2):1214-1229. https://doi.org/10.1109/TNSE.2019.2959230

Cederquist J, Corin R, Dashti MT, 2005. On the quest for impartiality: design and analysis of a fair non-repudiation protocol. Proc 7th Int Conf on Information and Communications Security, p.27-39. https://doi.org/10.1007/11602897_3

Chatterjee K, Raman V, 2014. Assume-guarantee synthesis for digital contract signing. *Formal Aspect Comput*, 26(4):825-859. https://doi.org/10.1007/s00165-013-0283-6

Christodoulou K, Christodoulou P, Zinonos Z, et al., 2020. Health information exchange with blockchain amid Covid-19-like pandemics. Proc 16th Int Conf on Distributed Computing in Sensor Systems, p.412-417. https://doi.org/10.1109/DCOSS49796.2020.00071

Deng HY, Hu RF, Pray C, et al., 2019. Impact of government policies on private R&D investment in agricultural biotechnology: evidence from chemical and pesticide firms in China. *Technol Forecast Soc Change*, 147:208-215. https://doi.org/10.1016/j.techfore.2019.07.011

Dinh TTA, Liu R, Zhang MH, et al., 2018. Untangling blockchain: a data processing view of blockchain systems. *IEEE Trans Know Data Eng*, 30(7):1366-1385. https://doi.org/10.1109/TKDE.2017.2781227

Erhan M, Tarhan A, Ozsoy A, 2019. A conceptual model for blockchain-based software project information sharing. Proc Joint Proc Int Workshop on Software Measurement and Int Conf on Software Process and Product Measurement, p.1-14.

Esfahani MM, Mohammed OA, 2018. Secure blockchain-based energy transaction framework in smart power systems. Proc 44th Annual Conf of the IEEE Industrial Electronics Society, p.260-264. https://doi.org/10.1109/IECON.2018.8591779

Feng XQ, Ma JF, Miao YB, et al., 2019. Pruneable sharding-based blockchain protocol. *Peer-to-Peer Netw Appl*, 12(4):934-950. https://doi.org/10.1007/s12083-018-0685-6

Griggs KN, Ossipova O, Kohlios CP, et al., 2018. Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. *J Med Syst*, 42(7):130. https://doi.org/10.1007/s10916-018-0982-x

Hang L, Kim DH, 2019. Design and implementation of an integrated IoT blockchain platform for sensing data integrity. *Sensors*, 19(10):2228. https://doi.org/10.3390/s19102228

Hansson H, Jonsson B, 1994. A logic for reasoning about time and reliability. *Formal Aspect Comput*, 6(5):512-535. https://doi.org/10.1007/BF01211866

Henzinger TA, Nicollin X, Sifakis J, et al., 1994. Symbolic model checking for real-time systems. *Inform Comput*, 111(2):193-244. https://doi.org/10.1006/inco.1994.1045

Kremer S, Markowitch O, 2000. Optimistic non-repudiable information exchange. Proc 21st Symp on Information Theory in the Benelux, p.139-146.

Kremer S, Markowitch O, Zhou JY, 2002. An intensive survey of fair non-repudiation protocols. *Comput Commun*, 25(17):1606-1621. https://doi.org/10.1016/S0140-3664(02)00049-X

Li TT, Ren W, Xiang YX, et al., 2021. FAPS: a fair, autonomous and privacy-preserving scheme for big data exchange based on oblivious transfer, ether cheque and smart contracts. *Inform Sci*, 544:469-484. https://doi.org/10.1016/j.ins.2020.08.116

Litchfield A, Herbert J, 2018. ReSOLV: applying cryptocurrency blockchain methods to enable global cross-platform software license validation. *Cryptography*, 2(2):10. https://doi.org/10.3390/cryptography2020010

Markowitch O, Roggeman Y, 1999. Probabilistic non-repudiation without trusted third party. Proc 2nd Conf on Security in Communication Networks, p.25-36.

Mendiboure L, Chalouf MA, Krief F, 2020. A scalable blockchain-based approach for authentication and access control in software defined vehicular networks. Proc 29th Int Conf on Computer Communications and Networks, p.1-11. https://doi.org/10.1109/ICCCN49398.2020.9209661

Mut-Puigserver M, Payeras-Capella MM, Cabot-Nadal MA, 2018. Blockchain-based fair certified notifications. Int Workshop on Cryptocurrencies and Blockchain Technology, p.20-37. https://doi.org/10.1007/978-3-030-00305-0_2

Novo O, 2018. Blockchain meets IoT: an architecture for scalable access management in IoT. *IEEE Internet Things J*, 5(2):1184-1195. https://doi.org/10.1109/JIOT.2018.2812239

Pérez-Solà C, Delgado-Segura S, Navarro-Arribas G, et al., 2019. Double-spending prevention for bitcoin zero-confirmation transactions. *Int J Inform Secur*, 18(4): 451-463. https://doi.org/10.1007/s10207-018-0422-4

Piramuthu S, 2017. RFID-based non-repudiation protocols for supply chains. Proc 3rd Int Conf on Future Network Systems and Security, p.56-69. https://doi.org/10.1007/978-3-319-65548-2_5

Resiere D, Resiere D, Kallel H, 2020. Implementation of medical and scientific cooperation in the Caribbean using blockchain technology in coronavirus (Covid-19) pandemics. *J Med Syst*, 44(7):123. https://doi.org/10.1007/s10916-020-01589-4

Roscoe AW, Ryan PYA, 2017. Auditable PAKEs: approaching fair exchange without a TTP. Proc 25th Cambridge Int Workshop on Security Protocols, p.278-297. https://doi.org/10.1007/978-3-319-71075-4_31

Ruland KC, Sassmannshausen J, 2015. Non-repudiation services for the MMS protocol of IEC 61850. Proc 2nd Int Conf on Research in Security Standardisation, p.70-85. https://doi.org/10.1007/978-3-319-27152-1_4

Said NB, Cristescu I, 2020. End-to-end information flow security for web services orchestration. *Sci Comput Programm*, 187:102376. https://doi.org/10.1016/j.scico.2019.102376

Sarr AP, Seye PB, Ngarenon T, 2019. A practical and insider secure signcryption with non-interactive non-repudiation. Proc 3rd Int Conf on Codes, Cryptology, and Information Security, p.409-429. https://doi.org/10.1007/978-3-030-16458-4_24

Wu JX, Gao Y, Zhang ZY, et al., 2018. A multi-party privacy preserving fair contract signing protocol based on blockchains. *J Cyber Secur*, 3(3):8-16 (in Chinese).

Yu KP, Tan L, Shang XL, et al., 2021. Efficient and privacy-preserving medical research support platform against COVID-19: a blockchain-based approach. *IEEE Consum Electron Mag*, 10(2):111-120. https://doi.org/10.1109/MCE.2020.3035520

Zhao YW, Sanán D, Zhang FY, et al., 2016. Reasoning about information flow security of separation kernels with channel-based communication. Proc 22nd Int Conf on Tools and Algorithms for the Construction and Analysis of Systems, p.791-810. https://doi.org/10.1007/978-3-662-49674-9_50

Zhou JY, Gollman D, 1996. A fair non-repudiation protocol. Proc IEEE Symp on Security and Privacy, p.55-61. https://doi.org/10.1109/SECPRI.1996.502669