Reputation-oriented Electronic Micro-loaning Based on Smart Contract in a Solidarity Group

Jung-San Lee[1], Ying-Chin Chen[1], Wei-Chih Hong[1], Kun-Yin Lee[2], Ren-Kai Yang[1]

[1] Department of Information Engineering and Computer Science, Feng-Chia University, Taiwan
[2] Industrial Ph.D. Program of Internet of Things, Feng-Chia University, Taiwan
leejs@fcu.edu.tw, ycchen.blythe@gmail.com,leo.jslee@gmail.com,
kunin19939368@gmail.com, mankindlee@gmail.com

Abstract

Digitalization apparently improves the efficiency and flexibility of financial services. In this work, we aim to introduce the Rotation Savings and Credit Association (ROSCA) into electronic commerce. ROSCA is one kind of non-interest lending and called as micro-loaning. The corresponding standards and rules are not severe restricted in comparison with those defined by financial institutions. It is friendly to people whose financial conditions are not so well. In particular, we adopt the smart contract which is a sort of automation technique to fulfill electronic micro-loaning in a solidarity group. Since all specifications of ROSCA are interpreted into functions via a logical transition, the contract content will be automatically implemented once the executing conditions are satisfied. This can significantly result in an effective operation and decrease the transaction cost. Moreover, the reputation strategy is applied to classify players into an appropriate group; thus, leading to mitigating the risk of embezzlement and insolvency.

Keywords: Micro-loaning, Smart contract, Reputation, e-commerce

1 Introduction

As to the explosive progress of electronic commerce (e-commerce), it undoubtedly brings the latest revolution of commerce to all. More than ninety percentages of traditional transactions could be switched to network platform, including trading, investment, insurance, and loan. It not only increases the turnover of merchant, but also provides more choices and convenience to people. The digital operation apparently improves the efficiency and flexibility of financial services [1-2]. Here we are going to introduce the Rotation Savings and Credit Association (ROSCA) into e-commerce. ROSCA is one kind of non-interest lending and called as micro-loaning. The corresponding standards and rules do not have such severe restrictions compared with that of financial institutions. It is friendly to people whose financial conditions are not well. Micro-loaning is executed by a small group during certain periods. A person who calls for initiating a micro-loaning is regarded as the originator. This game has several rounds according to the number of participants.

An example of micro-loaning is illustrated in Figure 1. There are five members, U_1, U_2, …, U_5. Suppose that the originator is U_1, the base prize is \$100, and the minimum interest/prize submitted from members is \$10. Each member only has one chance to be the round awardee (winner). After being the awardee, the participant has to pay the base prize \$100 in each round. Before becoming the winner, people hand in the money equal to the result of subtracting the highest interest of current session from the base. The first session is a non-interest lending, and U_1 is usually the awardee. That is, all members have to hand in \$100 to U_1. Suppose that U_2 is the awardee who offers the highest prize \$20 in the second round. Apart from U_1, the rest members have to pay \$80 to U_2. In case that U_3 spends \$25 to be the awardee in the third session, U_1 and U_2 have to pay the base prize to U_3 since they have been the round winner. The others offer U_3 \$75. This rotation awarding operation repeats until everyone has been the round winner once.

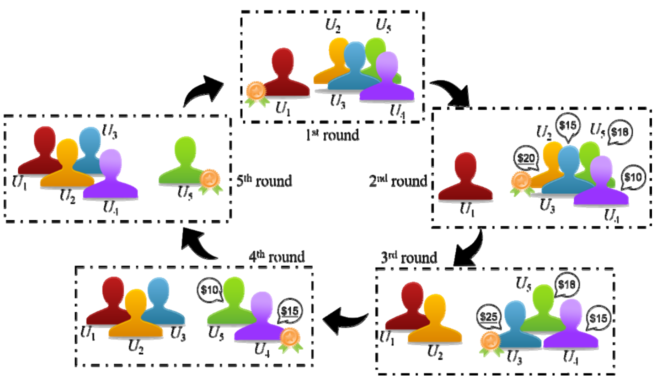


Figure 1. Micro-loaning example

Actually, an online peer-to-peer (P2P) lending

---

*Corresponding Author: Jung-San Lee; E-mail: leejs@fcu.edu.tw
DOI: 10.3966/160792642020072104016

mechanism is proposed in 2005. It is established between two individuals without a third party. Lacking bank charger makes debtor (borrower) and creditor (loaner) have a convenient lending circumstance. When the borrower intends to grant a loan, he/she first announces the loan purposes on the bulletin board of lending platform [3-4]. Next, the loaners bid the credit interest according to Dutch auction rules during the certain period. Finally, the system picks an awardee who submits the lowest one. In other words, the interest that is the most advantageous to borrower is selected. After that, the borrower has to regularly pay back to the loaner, including the capital and credit interest in the mortgage term. For debtor, the payable interest is lower than that of financial institution. Regards to the creditor, he/she earns greater profit compared with depositing money in the bank. The deficiency supervision of banking agency, however, easily causes the investment risk for loaner [5-8]. Also, P2P lending is still not kind enough to the needy. For example, suppose that a deprived person currently needs to borrow $10,000 for emergency. Under the equal requirements, he/she has to return more than $10,000 dollars in P2P lending because of the additional interest. On the contrary, the deprived person may repay lower than $10,000 dollars in a micro-loaning. It is because that other members might offer penny profit to earn the position of awardee. This could reduce the repayment pressure of the needy. In addition, the investment risk of creditor is eased since the lending is supported by a number of persons. This has motivated us to improve the loaning manner for offering the poverties a kindness-lending environment. Let people who have the similar finance level can assist each other over the Internet.

In this paper, we adopt the smart contract to fulfill electronic micro-loaning in a solidarity group. Smart contract is a sort of automation technique, and conventional specifications of ROSCA could be interpreted into functions via a logical transition. That is, the operation of smart contract has greatly matched to what it performs in an electronic micro-loaning. The contract content will be automatically implemented once the executing conditions are all satisfied. Consequently, this can significantly lead to an effective operation and decrease the cost consisting of human resource and working time. Rotation awarding diagram is shown in Figure 2. A user can request a micro-loaning initiation from bank through varied electronic devices. The information set up by the originator is applied to launch the smart contract. The micro-loaning is implemented when a pre-defined number of members have joined. Next, participants submit the prize during rotation awarding period. After that, the round winner will be automatically selected on the session deadline. The bank subsequently complies the wire transfer. The smart contract keeps running until each member has experienced the round awardee.

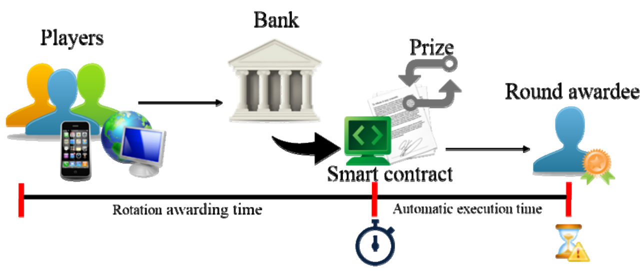


Figure 2. Rotation awarding diagram of the proposed scheme

In particular, the reputation strategy is imported to mitigate investment risk such as embezzled loan, in which a participant may refuse to hand in payment after being round awardee. Moreover, we bring in the role of bank to carry out an effective transaction monitoring. This could lower down the distrust problem of an electronic micro-loaning. Thus, the proposed mechanism is able to provide preferable circumstance for lending market having a balance evolvement. Specifically, the security of the proposed scheme is authenticated by the formal tool, Automated Validation of Internet Security and Applications (AVISPA) which is widely approved by standard institutions [9].

The rest of this article is organized as follows. The related work is introduced in Section 2. The proposed scheme is explained in Section 3, followed by analysis & discussion and implementation results in Sections 4 and 5, respectively. The formal verification using AVISPA is shown in Section 6. Finally, we make conclusions in Section 7.

2 Related Work

Here, we describe the scenario of a smart contract and define the requirements in more details. Nick Szabo first proposed the idea of smart contract which contains the concept of physical contract in 1996 [10-12]. The general definition of physical contract is a legal binding agreement. The content has to be consented by the parties. Afterwards, the parties can achieve the valid property exchange. The contracts have been widely applied to trading, ownership transfer, and engagement in specific written or oral agreements. However, people spend a large number of human resource maintaining and implementing the physical contract. Therefore, the smart contract terms are described in a logical program. When all the conditions are satisfied, the predefined functions are automatically performed. For example, the vendor machine will automatically supply a bottle of soda while the buyer puts a dollar and presses the chosen button. The purchasing behaviors are displayed in Figure 3.

```
Algorithm
Input: money, button
Output: soda
contract Vendor_Machine{
    soda_price = 1
    received_money = 0
    button = False

    function buying(money, button){
        if (money == 1 && button == True){
            release_soda() }
}
```

Figure 3. Pseudo code

The definition of smart contract is illustrated in a white paper designed by Szabo in 2016. It consists of four elements: promise, digital form, protocol, and automation [10-12].

(1) *Promise*

It is a set of promises referred to the contractual terms of an agreement between the joined parties, including mandatory obligations and enjoyment of right. The value and aim of contract are defined in the promise. For example, in a business dealing, if two parties intend to establish a legal contract, the buyer needs to pay the reasonable price, and the seller has to promise products delivering.

(2) *Digital form*

A smart contract executes in electronically way, consisting of a bunch of code. Each contractual term is written as a function. When two parties have negotiated an agreement, the rights and obligations of smart contract are performed through the Internet and computers.

(3) *Protocol*

The electronic protocol is a set of rules which are depicted in an algorithm form. The functions of protocol define how to deal with the information according to smart contract. In addition, these manipulations are executed via technology-enable and rules-based operations, such as the effect payment.

(4) *Automation*

Automatic execution is the kernel of smart contract. When the smart contract becomes effective, it voluntarily conducts the operations except for the unsatisfied conditions. Namely, no one can stop it in the technique way.

3 The Proposed Scheme

In this section, we depict how the smart contract and reputation strategy operate in an electronic micro-loaning. It includes four phases: registration phase, initialization phase, rotation awarding phase, and reputation update phase. The used notations are shown in Table 1.

Table 1. Notations

Sign	Definition
SC	Smart contract
U_i	A general player (user)
$Bank$	Smart contract generator
x	The master key of *Bank*
PD	Participation deadline
TH	Reputation threshold
SN	The number of times originator has launched
$\oplus$	Exclusive-or operation (XOR)
$\|\|$	Concatenation symbol
rep_i	Reputation value of U_i
GK	Group-key used in rotation awarding
acc_i	Account of U_i
LM	Micro-loaning message
n	The number of participants
B	The base prize
id_i/pw_i	Identity/Password of U_i
T	Timestamp
$E_k/D_k[.]$	Encryption/Decryption using key k
$h(.)$	One-way hash function

3.1 Registration Phase

Before a player participates a micro-loaning, he/she has to register at the bank via a secure channel as shown in Figure 4.

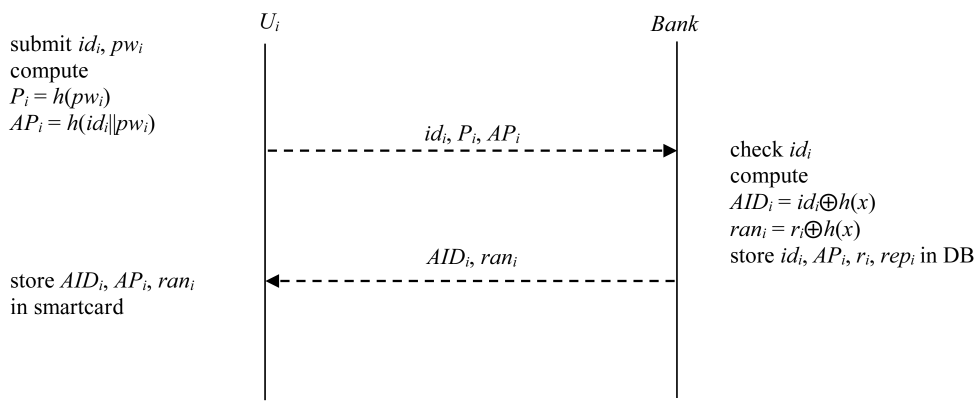


Figure 4. Registration phase

3.2 Initialization Phase

This phase includes login & authentication procedure so that U_i has the ability to initialize and join a micro-loaning. After registration phase, U_i is able to login to *Bank*, and the session key SK_i will be negotiated for micro-loaning in this phase. Later on, if a player intends to launch a micro-loaning, U_i has to initialize the scenario setting through the following steps. Suppose that the originator of a micro-loaning is U_1. Only U_1 needs to set up the conditions as shown in step 4. Afterwards, other members can attend the micro-loaning according to *SCSN* as shown in step 6. Also, the group-key *GK* used in rotation awarding phase is established in this part. The diagram is displayed in Figure 5.

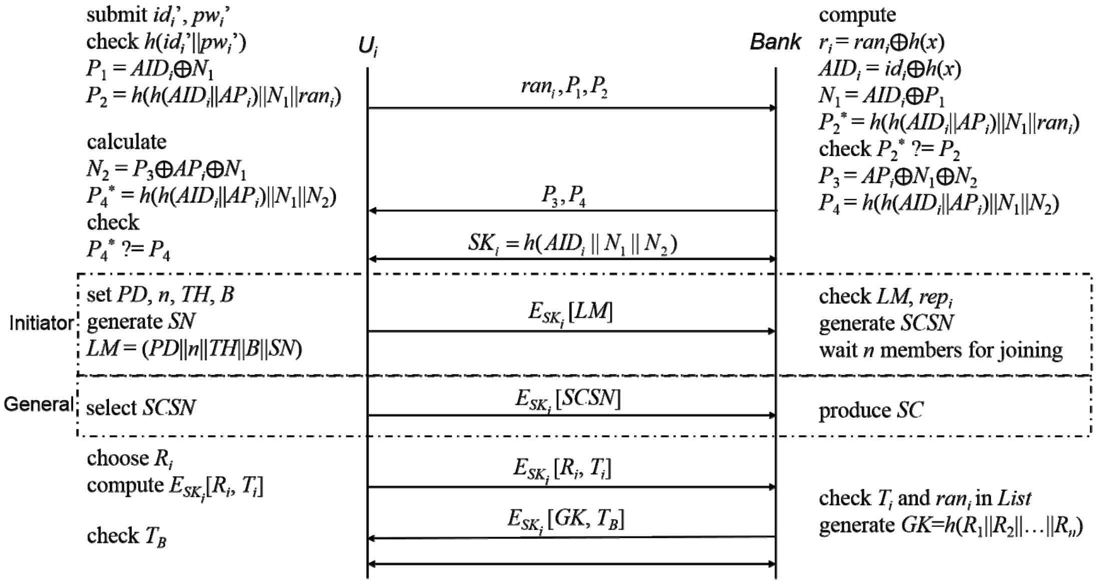


Figure 5. Initialization phase

Step 1: U_i enters id_i' and pw_i' into smart device for checking whether $h(id_i'||pw_i')$?= AP_i or not. If it holds, smart card computes the security parameters $P_1 = AID_i \oplus N_1$ and $P_2 = h(h(AID_i||AP_i)||N_1||ran_i)$. Later, the login request $\{ran_i, P_1, P_2\}$ is transmitted to *Bank*.

Step 2: Upon obtaining the request, *Bank* reveals four parameters $r_i = ran_i \oplus h(x)$, $AID_i = id_i \oplus h(x)$, $N_1 = AID_i \oplus P_1$, and $P_2^* = h(h(AID_i||AP_i)||N_1||ran_i)$. After that, *Bank* verifies whether P_2^* ?= P_2. If it is valid, *Bank* generates a random nonce N_2 and calculates $P_3 = AP_i \oplus N_1 \oplus N_2$ and $P_4 = h(h(AID_i||AP_i)||N_1||N_2)$. Finally, the login response $\{P_3, P_4\}$ is sent back to U_i.

Step 3: Once accepting the response, U_i discloses N_2 from P_3 and computes $P_4^* = h(h(AID_i||AP_i)||N_1||N_2)$. If P_4^* is equal to P_4, the session key SK_i is successfully constructed, where $SK_i = h(AID_i||N_1||N_2)$.

Step 4: U_1 selects the conditional values and generates the micro-loaning message $LM = (PD||n||TH||B||SN)$. Next, U_1 applies SK_i to encrypt and send *LM* to *Bank*.

Step 5: Upon decrypting and acquiring *LM*, *Bank* first checks whether *LM* has been established or not. If the *LM* is fresh, *Bank* generates a unique number *SCSN* for micro-loaning and announces to bulletin board for other members inquiring. Afterwards, *Bank* waits until the expired period or the number of members is reached.

Step 6: The rest users U_i send the join request $\{E_{SK_i}[SCSN]\}$ to *Bank*.

Step 7: When the waiting time is up, *Bank* first examines whether the reputation of each member in list meets *TH*. If all the values are effective, *Bank* sequentially produces the corresponding smart contract *SC* which is used to control a micro-loaning. *SC* is able to automatically generate *n* Sub-SC_is' according to the number of joiners. Sub-SC_1 is built along with *SC* construction since the micro-loaning is started up. Moreover, the states of *SC* and Sub-SC_1 are standby until the rotation awarding is executed.

Step 8: When the micro-loaning has been successfully launched, each member U_i in the list needs to choose a random number R_i for group-key generation. The values R_i and T_i of each U_i are encrypted with SK_i which is established between U_i and *Bank*, respectively. Later, U_i sends the security message $\{E_{SK_i}[R_i, T_i]\}$ to *Bank* for generating *GK*.

Step 9: After obtaining the security message from U_i, *Bank* separately checks whether T_i is fresh or not and if member U_i is in list according to ran_i. If both values are correct, *Bank* keeps R_i until the expiration date. Next, *Bank* constructs $GK = h(R_1||R_2||\ldots||R_n)$. Finally, *Bank* separately transmits $\{GK, T_B\}$ to U_i with an individual session key.

Step 10: When receiving the response during the valid period, U_i uses the SK_i to decrypt the security message and checks the timestamp. If it is fresh, *GK* is accepted in this micro-loaning round. Otherwise, the initialization is failed.

3.3 Rotation Awarding Phase

After successfully establishing the group-key for the micro-loaning, U_i starts to award during the valid period. Before U_i submits the prize in each awarding round, U_i has to pass the login and authentication process as shown in steps 1-3 of section 3.2. Each individual micro-loaning has a *SC* and *n* Sub-SC_is'. The diagram is displayed in Figure 6.

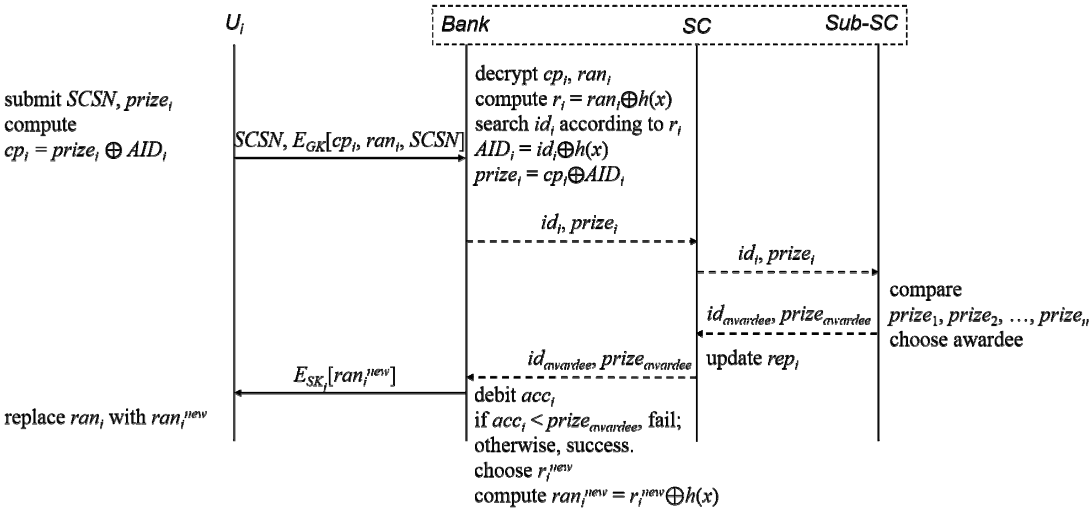


Figure 6. Rotation awarding phase

Step 1: U_i computes $cp_i = prize_i \oplus AID_i$ and transmits $\{SCSN, E_{GK}[cp_i, ran_i, SCSN]\}$ to *Bank*. U_i is able to submit the $prize_i$ during the valid period, or the default prize is presented.

Step 2: After accepting the message, *Bank* discloses cp_i and ran_i, and *SCSN* according to *GK*. Next, *Bank* reveals $prize_i$ and employs ran_i to search the corresponding id_i from its database and forwards the result and $prize_i$ to related *SC* based on *SCSN*.

Step 3: Upon receiving the input values, the state of *SC* is converted into active. Then, *SC* directly transmits id_i and $prize_i$ to Sub-SC_i. Once the availability period of prize submitting expires, Sub-SC_i automatically launches the execution for comparing all prizes to choose an awardee. Finally, the award prize and awardee $\{prize_{awardee}, id_{awardee}\}$ are sent back to *SC* for being recorded.

Step 4: After acquiring the parameters from Sub-SC_i, $\{prize_{awardee}, id_{awardee}\}$ are announced and recorded in *SC*. The value rep_i of each U_i in list is counted in its queue for the last summation. Ultimately, $\{prize_{awardee}, id_{awarde}\}$ are passed to *Bank* for financial actions, such as transferring and debiting.

Step 5: When obtaining the $prize_{awardee}$, $id_{awardee}$ from *SC*, *Bank* checks whether the deposit of each U_i is more than $prize_{awardee}$ in the valid time. If any bank account of U_i is lower than $prize_{awardee}$, the awarding is failed. Otherwise, *Bank* successfully transfers the amount $prize_{awardee} \times (n\text{-}1)$ from non-awardees U_i to $U_{awardee}$. At last, *Bank* selects different random nonce r_i^{new} and computes distinct $ran_i^{new} = r_i^{new} \oplus h(x)$ for each U_i to renew pseudonym identity. The *n* individual ran_i^{new} are respectively delivered to each U_i from *Bank* using separate session key.

Step 6: Once accepting the response, U_i decrypts message and replaces ran_i with ran_i^{new}.

Step 7: Each awarding round is executed as step 1 to step 6 until the *n*th awarding has been completed.

3.4 Reputation Update Phase

After micro-loaning playing off, *SC* sums up and conveys rep_i of each member to *Bank*. Later on, *Bank* updates the rep_i in database according to id_i.

4 Analysis and Discussion

Here we analyze the essential requirements of a secure protocol in subsection 4.1, followed by the discussion on achieved functionalities in subsection 4.2.

4.1 Essential Requirement Analysis

Regarding to the e-commerce requirements, we focus on mutual authentication, perfect forward secrecy, anonymity, and confidentiality. The assumptions are depicted in the following.

(1) Symmetric encryption function $[.]_K$

It is impractical to decrypt message $[M]_K$ without having the secret key *K* [13].

(2) One-way hash function *h*(.)

Based on [14], one-way hash function *h*(.) will return a fixed size output even if the input is a changeable length message. There are three principles of *h*(.) listed in the following.

Pre-image resistance: Supposed that a string of information $y = h(M)$, it is unavailable to learn the original information *M* from *y*.

The second pre-image resistance: Given the message digest $y = h(M)$ and the message M, it is impossible to find another message M' to meet the same output $y = h(M')$.

(3) Exclusive-or operation $\oplus$

Supposed that the ciphertext C is computed by messages M_1 and M_2 using exclusive-or operation. It is computationally infeasible to derive C without either M_1 or M_2 in polynomial time. However, it is easily to obtain $C = M_1 \oplus M_2$.

4.1.1 Mutual Authentication

Mutual authentication means that both the validities of player and bank shall be verified by each other in initialization phase. Once receiving the request $\{ran_i, P_1, P_2\}$ from U_i, *Bank* first uses $h(x)$ to reveal r_i and checks whether the corresponding id_i has existed or not. If so, *Bank* sequentially calculates $AID_i = id_i \oplus h(x)$, $N_1 = AID_i \oplus P_1$, and $P_2^* = h(h(AID_i||AP_i)||N_1||ran_i)$. Furthermore, *Bank* verifies U_i according to P_2^* ?= P_2. If they are the same, U_i is checked by *Bank*. It is due to the fact that only the valid *Bank* can filter r_i and AID_i using $h(x)$ under the assumption of exclusive-or operation. As a result, it represents that U_i is a valid user, and the login request is accepted by *Bank*.

On the other hand, upon acquiring the response, U_i utilizes $AP_i = h(id_i||pw_i)$ and N_1 to disclose N_2 from P_3. Next, U_i computes $P_4^* = h(h(AID_i||AP_i)||N_1||N_2)$ and checks whether P_4^* ?= P_4. If they are equal, *Bank* is authenticated by U_i. It is because that only the legal U_i is able to get N_2 using N_1 and AP_i based on the assumption of exclusive-or operation. Therefore, U_i and *Bank* can mutually authenticate each other and establish the common session key $SK_i = h(AID_i||N_1||N_2)$ in the proposed mechanism.

4.1.2 Perfect Forward Secrecy

Suppose that an adversary *Ivy* intends to construct the previous session key and the future one using the present transmitted messages. However, she will fail to achieve the goal since she cannot compute AID_i without id_i and $h(x)$ based on XOR operation. Even she can obtain AID, N_1, and N_2, SK_i is unworkable. It is because that N_1 and N_2 of SK_i are random nonces in each login session. Thus, our scheme can provide the feature of perfect forward secrecy.

4.1.3 Anonymity

Each player has an individual identity according to the environment setting. Due to the fact that id_i of U_i is not transmitted via the Internet, only the legitimate *Bank* has ability to trace a specific U_i. The message ran_i is applied to protect the value r_i which is used to recognize a specific U_i and is delivered from U_i to *Bank* during initialization process. If an adversary *Ivy* has intercepted initialization requests and intends to track a certain U_i, she must fail to reveal the value r_i from ran_i without $h(x)$ under the assumption of exclusive-or operation. Furthermore, the value ran_i is updated after each rotation awarding. Even *Ivy* obtains the previous ran_i, the value is useless. As to the rotation awarding procedure, U_i sends awarding request $\{SCSN, E_{GK}[cp_i, ran_i, SCSN]\}$ to *Bank*, which contains ran_i. It is computationally infeasible for *Ivy* to decrypt secret message without GK based on the symmetric cryptosystem. Therefore, the proposed scheme can confirm the property of anonymity.

4.1.4 Confidentiality

The message, which consists the sensitive information, is secure transmitted during each session. These secret values include id_i and pw_i of U_i, N_1 and N_2 for generating session key, and R_i for producing group-key. Here we suppose that an attacker *Ivy* is able to intercept the delivered message. In the initialization phase, U_i sends $\{ran_i, P_1, P_2\}$ to *Bank*. The value ran_i contains r_i which is a unique index of U_i issued by *Bank* during the registration procedure. However, under the setting of exclusive-or operation, it is computationally infeasible for *Ivy* to reveal r_i without $h(x)$ of *Bank*. Next, the parameter P_1 retains N_1. Only the proper *Bank* has correct value AID_i to reveal N_1. On the other hand, *Bank* responses $\{P_3, P_4\}$ to U_i. The variable P_3 owns the secret data N_1 and N_2. Whereas, *Ivy* has no idea about acquiring N_1 and N_2 since only the valid U_i has ability to compute AP_i under the assumption of XOR operation. Afterwards, U_i passes $\{E_{SK_i}[R_i, T_i]\}$ to *Bank* for producing group-key. If an attacker *Ivy* intends to reveal R_i, she has to provide SK_i to decrypt transmitted message. However, she cannot achieve the goal since only the valid U_i and *Bank* own the correct SK_i based on symmetric cryptosystem. Hence, she cannot learn any confidential information during the initialization phase.

Regarding to rotation awarding phase, U_i delivers awarding request $\{SCSN, E_{GK}[cp_i, ran_i, SCSN]\}$ to *Bank*, which contains sensitive information cp_i and ran_i. If a malicious attacker *Ivy* wants to filter the parameters, she has to offer the correct GK. Nevertheless, it is computationally infeasible for her to generate a valid GK without joining the initialization phase under the assumption of symmetric cryptosystem. Consequently, the proposed mechanism is able to ensure the confidentiality of secret message during each session.

4.2 Functionality Discussion

We probe the functions of proposed mechanism in two aspects: Topic 1: why does the micro-loaning scheme correspond with the definitions of smart contract and what kinds of feature does it possess? Topic 2: why do we introduce the reputation rationale to propose a trust micro-loaning?

Topic 1: Here we offer the explanation to prove that this article satisfies the definitions in subsections 2.1-2.4.

(1) *Promise:* A set of promises refer to the contractual terms of an agreement between the joined parties. Before launching a micro-loaning, the originator U_1 has to set up fundamental requirements, including *PD*, *n*, *TH*, and *B*. These conditions are considered as the promises of the micro-loaning. When the player U_i intends to join a micro-loaning, U_i has to accept the agreement. Therefore, the proposed mechanism meets the first condition in subsection 2.1.

(2) *Digital form*: A smart contract executes in an electronic way. In the proposed scheme, each user is able to access the platform through the Internet by distinct devices. Besides, the originator is not directly responsible to convey the message and comply the campaign in a micro-loaning. Also, the physical records are substituted with digital ones. It is good to conserve documents for a long time. Consequently, the second condition in subsection 2.2 is satisfied.

(3) *Protocol:* The electronic protocol is a set of rules which are depicted in an algorithm form. According to analysis in section 4.1, the smart contract of our method is appropriate controlled by function code which is described on the basis of contract terms. Therefore, our scheme can firmly preserve the confidentiality, integrity, and availability of input through the Internet. As a result, the third condition in subsection 2.3 is reached.

(4) *Automationv:* When the smart contract becomes effective, the system voluntarily conducts the operations except for the unsatisfied conditions. The proposed scheme has two smart contract types for managing operations at a particular time, *SC* and Sub-SC_i. Each individual micro-loaning has a *SC* and *n* Sub-SC_is' as mentioned in sections 3.2 and 3.3. *SC* is the main contract used to control a micro-loaning, such as essential settings and records. Sub-SC_i is utilized to execute each awarding round in the third phase for automatically handling the time break point. During the rotation awarding phase, the originator does not gather the payment by himself/herself any more. Sub-SC_i is able to automatically compare the round prize of each participant and then the round awardee is selected. After that, *SC* transmits the information to *Bank* who has ability to collect the payment from non-awardees and transfer money to the round awardee. This can reduce the personnel cost and improve the finance usage. Hence, the fourth condition in subsection 2.4 is accomplished.

Topic 2: Reputation mechanism is utilized to construct the trust foundation of members. The micro-loaning is divided into ten levels according to reputation value as shown in Table 2. The standards are defined by consumer habitat and currency domination. Undoubtedly, the standards are able to be adjusted in different nations and specific regions. The reputation value is increased three units after each successful session. For a player, the number of available lending is determined by its reputation. The total participated loan shall be lower than the personal reputation value, and the maximum of joined games is three. This can offer a stable circumstance and mitigate the burden of poverty. Moreover, the unit of a round period is one month since the salary is generally month-pay. The cycle of a micro-loaning is classified in three classes: a season, six months, and a year, in which the number of joined members is three, six, and twelve, respectively. Note that the reputation value of a client is returned to default one when insolvency problem happened. If one person does deception again, he/she will be put in blacklist.

Table 2. Example of prize level

Type	Reputation (unit)	Base ($)	The minimum awarding prize ($)
A	10	1,000	100
B	20	2,000	100
C	30	3,000	100
D	40	4,000	100
E	50	5,000	100
F	60	6,000	100
G	70	7,000	100
H	80	8,000	100
I	90	9,000	100
J	100	10,000	100

For example, supposed that a user *Alice* has 22 units of reputation so that she has multiple choices for lending combination. She can select one micro-loaning of type A, two type A, or one type B during a certain period. If she participates one type B loan for six months, she might acquire $12,000 in the first round. Next, she pays back lower than or equal to the base prize in the rest of episodes. Based on the reputation management, players can join a suitable game to acquire appropriate support and avoid the risk of insolvency. Actually, all these settings could be tuned according to the circumstance.

5 Implementation Results

In this section, we implement the proposed scheme under the environment of Python 3.6.0. The outcomes of communication overhead and computation cost in our scheme are separately described in section 5.1 and section 5.2. A personal computer is installed with Windows 10 64-bits for simulating the micro-loaning environment, equipped with Intel-Core i5-7400 3.0 GHz and 8 GB RAM. The assumptions are depicted as follows. The lengths of id_i, pw_i, random nonce, and timestamp are 32 bits. The size of x is 256 bits. The version of AES and one-way hash function are AES-256 and SHA-256, respectively. Here we select twelve players to participate a micro-loaning during the simulation.

5.1 Communication Overhead

Communication overhead represents that the total transmitted data during each phase, including player and bank. The details are displayed in Table 3. In the registration phase, id_i, P_i, and AP_i are transmitted from U_i to *Bank*. id_i is assumed to be 32 bits. As $P_i = h(pw_i)$ and $AP_i = h(id_i||pw_i)$, P_i and AP_i are 256 bits. Therefore, the cost of U_i is 32+256+256 = 544 bits. The cost for *Bank* is 512 bits since *Bank* sends AID_i and ran_i to U_i, where $AID_i = id_i \oplus h(x)$ and $ran_i = r_i \oplus h(x)$. During the initialization phase, it contains the processes of login & authentication and micro-loaning. U_i sends ran_i, P_1, and P_2 to *Bank*. The length of ran_i is 256 bits stored in smart card. P_1 is the XOR operation outcome formed by AID_i and N_1. P_2 is $h(h(AID_i||AP_i)||N_1||ran_i)$. Therefore, both the lengths of P_1 and P_2 are 256 bits. The cost of U_i is 256+256+256 = 768 bits. *Bank* responses P_3 and P_4 to U_i, where $P_3 = AP_i \oplus N_1 \oplus N_2$ and $P_4 = h(h(AID_i||AP_i)||N_1||N_2)$. The cost of *Bank* is 256+256 = 512 bits.

Table 3. Communication overhead (bit)

		Player		Bank
Registration phase		544		512
Initialization phase	Login & Authentication part	768		512
	Micro-loaning part	Originator	General	4608
		384	256	
Rotation awarding phase		672		256
Total		Originator	General	5888
		2368	2240	

Regarding to the micro-loaning phase in initialization procedure, we first assume that the initial values of *PD*, *TH*, *B*, *SN*, R_i, and T_i are all of 32 bits. The communication overheads of player are divided into two types, the originator U_1 and general users U_2, U_3, …, U_{12}. U_1 first passes the E_{SK_1} [*LM*] and E_{SK_1} [R_1, T_1] to *Bank*. The size of encrypted *LM* is 256 bits since it takes two AES cipher blocks. As to the length of secret pair (R_1, T_1), it is 128 bits for one AES cipher block. Therefore, the cost of U_1 is 256+128 = 384 bits. Later on, the other player only sends E_{SK_1} [*SCSN*] and E_{SK_1} [R_i, T_i] to *Bank*. Hence, the cost of a general player is 128+128 = 256 bits. Next, since *Bank* separately transmits {*GK*, T_B} to each U_i with distinct session keys, the overhead of *Bank* is 384 × 12 = 4608 bits.

As to the rotation awarding phase, U_i conveys *SCSN* and E_{GK} [cp_i, ran_i, *SCSN*] to *Bank*. E_{GK} [cp_i, ran_i, *SCSN*] is 640 bits since the encryption of cp_i, ran_i, and *SCSN* required five AES cipher blocks. Thus, the cost of U_i is 32+640 = 672 bits. *Bank* only encrypts ran_i^{new} to U_i using SK_i. The load for *Bank* is 256 bits. The total burden of originator U_1, normal players U_2-U_{12}, and *Bank* are 2368 bits, 2240 bits, and 5888 bits, respectively.

Above-mentioned calculations have displayed the fact that all participants in the play need not to pay heavy bit lengths to complete the transaction. That is, the communication overhead is light to be practically employed in mobile devices.

5.2 Computation Cost

Computation costs have involved all the operations performed by players and bank during each phase. $T_{E_{AES}}$, $T_{D_{AES}}$, T_H, and T_{XOR} denote the execution time of one AES encryption, AES decryption, one-way hash function, and exclusive-or operation, respectively. To avoid the existence of extreme value, we have conducted the simulation one million times for each procedure to obtain the average execution time. The computation cost of each phase is illustrated in Table 4. As to the registration phase, U_i only executes two T_H operations. The time cost of U_i is 0.0023 ms. *Bank* spends 0.0125 ms performing one T_H and two T_{XOR} operations.

During the login & authentication part of initialization phase, each U_i computes six T_H and three T_{XOR} operations. *Bank* deals with six T_H and five T_{XOR} operations. In the micro-loaning procedure, the computation cost is also classified in two types, the originator U_1 and general players U_2, U_3, …, U_{12}. The time consumption of U_1 is 0.3642 ms, including two $T_{E_{AES}}$ and one $T_{D_{AES}}$ operations. On the other hand, the normal players, U_2-U_{12}, spend 0.3641 ms executing two $T_{E_{AES}}$ and one $T_{D_{AES}}$. *Bank* consumes 2.2138 ms to apply twelve $T_{E_{AES}}$, 24 $T_{D_{AES}}$, and one T_H.

In the terms of rotation awarding period, it takes U_i 0.1773 ms to handle one $T_{E_{AES}}$, one $T_{D_{AES}}$, and one T_{XOR}. *Bank* runs twelve $T_{E_{AES}}$, twelve $T_{D_{AES}}$, twelve T_H, and 48 T_{XOR} for 2.5820 ms. The total cost of U_1, U_2-U_{12}, and *Bank* are 0.5445 ms, 0.5444 ms, and 4.8083 ms, respectively.

Table 4. Computation cost (ms)

		Player		Bank
Registration phase		$2T_H$		$1T_H+2T_{XOR}$
		0.0023		0.0125
Initialization phase	Login & authentication part	$6T_H+3T_{XOR}$		$6T_H+5T_{XOR}$
	Micro-loaning part	Originator	General	$12T_{E_{AES}}+24T_{D_{AES}}+1T_H$
		$2T_{E_{AES}}+1T_{D_{AES}}$	$2T_{E_{AES}}+1T_{D_{AES}}$	
		0.3642	0.3641	2.2138
Rotation awarding phase		$1T_{E_{AES}}+1T_{D_{AES}}+1T_{XOR}$		$12T_{E_{AES}}+12T_{D_{AES}}+12T_H+48T_{XOR}$
		0.1773		2.5820
Total		Originator	General	$24T_{E_{AES}}+36T_{D_{AES}}+20T_H+55T_{XOR}$
		$3T_{E_{AES}}+2T_{D_{AES}}+8T_H+4T_{XOR}$	$3T_{E_{AES}}+2T_{D_{AES}}+8T_H+4T_{XOR}$	
		0.5445	0.5444	4.8083

It is clear that each step to secure the trasnaction can be done quickly no matter it happens to the player or the bank. Consequently, the proposed scheme has possessed the feature of efficiency.

6 Formal Verification Using AVISPA

To highlight the secuirty of the proposed mechanism, we examine its robustness through a formal tool, Automated Validation of Internet Security Protocols and Applications (AVISPA) [15].

There are integrated four back-end validators in AVISPA, consisting of On-the-Fly-Model-Checker (OFMC), Constraint-Logic-based Attack Searcher (CL-ATSE), Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP), and SAT-based Model-Checker (SATMC). AVISPA is utilized to check whether the proposed scheme suffers from network attacks or has any vulnerability. The version of AVISPA for simulation is the Security Protocol Animator version 1.6 (SPAN 1.6) installed on an Ubuntu 10.10 light workstation with an Intel-Core i7-3770K CPU running at 3.50 GHz with 1.00 GB of RAM.

High Level Protocol Specification Language (HLPSL) is adopted in AVISPA to depict two element roles for our scheme, player and bank. Also, the intruder knowledge is described in the protocol. The parameters of a player and a bank in the proposed scheme are illustrated in HLPSL specification as shown in Figure 7. After that, we define the session and environment in HLPSL to conduct the initiation for the element roles as shown in Figure 8. Table 5 is cross reference chart between the notations used in this article and specification parameters in HLPSL. The Dolev-Yao model is applied in simulation for intruder knowledge [16]. We further apply the strong authentication features in N_1, N_2, *ran*, *R*, and *prize* for examining replay attack, impersonation attack, and server spoofing attack in Figure 9.

Here, CL-ATSE and OFMC are utilized to analyze our scheme since the back-end validators, TA4SP and SATMC, do not support the property of exponential operation. The approach applying to intermediate format of CL-ATSE is opposed to OFMC. The iterated sessions and loops have been processed in CL-ATE due to integrating the optimized Baader & Schulz unification algorithm [17]. On the other hand, the modularization is adopted in OFMC to deal with the procedure [18]. The outcome of OFMC is safe as shown in Figure 10. In the terms of CL-ATSE, we adopt two models and one mode to conduct our scheme, typed model, untyped model, and verbose mode, respectively. The whole types of parameters are referred to typed model. On the contrary, each type of values has to be transferred to generic type in untyped model. About verbose mode, the detail of intruder trace is described when the protocol is insecure. The experimental consequences in Figure 11 have displayed that our scheme is secure against any Internet attack.

7 Conclusions

The contribution of this work can be summarized as follows. The adoption of smart contract can efficiently help to enhance the performance of each transaction over the Internet. Furthermore, the security of the whole system has been guaranteed according to AVISPA which is a famous tool to examine the robustness of authentication protocol. In the future, we aim to improve the reputation strategy to comply with the realistic network financial environment.

Player

```
role player (
    U, B: agent,
    Ku, Kb: public_key,
    Hash: hash,
    SND, RCV: channel(dy)
)
played_by U def=
local
    State: nat,
    Id, Pw, P, Ap, Ran, Aid, P1, P2, P3, P4: text
    R1, N1, N2, Rn, T1, Tb, Prize: text,
    Sk, Gk: message
const ran, sk, gk, bank_player_ran, player_bank_n1, bank_player_n2, player_bank_rn, player_bank_prize:
protocol_id
init State := 1
transition
0.  State = 0 ∧ RCV(start) =|> State': = 2
            ∧ Id': = new ()
            ∧ Pw': = new ()
            ∧ P': = Hash(Pw')
            ∧ Ap': = Hash (Id'. Pw')
            ∧ SND ({Id'. P'. Ap'}_Kb)
2.  State = 2 ∧ RCV ({Aid'. Ran'}_Ku) =|> State':= 4
            ∧ N1': = new ()
            ∧ P1': = xor (Aid', N1')
            ∧ P2': = Hash (Hash (Aid'. Ap). N1'. Ran')
            ∧ SND (Ran', P1', P2')
            ∧ request (U, B, bank_player_ran, Ran')
            ∧ witness (U, B, bank_player_ran, Ran')
            ∧ witness (U, B, player_bank_n1, N1')
4.  State = 4 ∧ RCV (P3', P4') =|> State': = 6
            ∧ request (U, B, bank_player_n2, N2')
            ∧ N2': = xor (P3', xor (Ap, N1))
            ∧ P4': = Hash (Hash (Aid', Ap, N1, N2'))
            ∧ Sk': = Hash (Aid'. N1. N2')
            ∧ secret (Sk', sk, {U, B})
            ∧ Rn': = new ()
            ∧ T1':= new()
            ∧ SND ({Rn'. T1'}_Sk', Ran')
            ∧ witness (U, B, bank_player_ran, Ran')
            ∧ witness (U, B, player_bank_rn, Rn')
6.  State = 6 ∧ RCV ({Gk'. Tb'}_Sk') =|> State': = 8

            ∧ SND ({Prize'. Ran'}_Gk')
            ∧ witness (U, B, player_bank_prize, Prize')
end role
```

Bank

```
role bank (
    B, U: agent,
    Kb, Ku: public_key,
    Hash: hash,
    SND, RCV: channel(dy)
)
played_by B def=
local
    State: nat,
    Id, Pw, P, Ap, Ran, Aid, P1, P2, P3, P4: text
    R1, N1, N2, Rn, T1, Tb, Prize: text,
    Sk, Gk: message
const ran, sk, gk, bank_player_ran, player_bank_n1, bank_player_n2, player_bank_rn, player_bank_prize:
protocol_id
init State: = 0
transition
1.  State = 1 ∧ RCV ({Id'. P'. Ap'}_Kb) =|> State': = 3
            ∧ Aid': = xor (Id', Hash (inv (Kb)))
            ∧ R1' : = new ()
            ∧ Ran': = xor (R1', Hash (inv (Kb)))
            ∧ SND ({Aid'. Ran'}_Ku)
            ∧ secret (Ran', ran, {U, B})
            ∧ witness (B, U, bank_player_ran, Ran')
3.  State = 3 ∧ RCV (Ran, P1', P2') =|> State': = 5
            ∧ request (B, U, bank_player_ran, Ran)
            ∧ request (B, U, player_bank_n1, N1')
            ∧ R1': = xor (Ran, Hash (inv (Kb)))
            ∧ Aid': = xor (Id', Hash (inv (Kb)))
            ∧ N1': = xor (Aid, P1')
            ∧ P2': = Hash (Hash (Aid. Ap'). N1'. Ran')
            ∧ N2': = new ()
            ∧ P3': = xor (Ap', xor (N1'. N2'))
            ∧ P4': = Hash (Hash (Aid. Ap'). N1'. N2')
            ∧ SND (P3', P4')
            ∧ witness (B, U, bank_player_n2, N2')
            ∧ Sk': = Hash (Aid. N1'. N2')
            ∧ secret (Sk', sk, {U, B})
5.  State = 5 ∧ RCV ({Rn'. T1'}_Sk', Ran') =|> State': = 7
            ∧ request (B, U, bank_player_ran, Ran)
            ∧ request (B, U, player_bank_rn, Rn')
            ∧ Gk': = Hash (Rn')
            ∧ Tb': = new ()
            ∧ SND ({Gk'. Tb'}_Sk')
            ∧ secret (Gk', gk, {U, B})
7.  State = 7 ∧ RCV ({Prize'. Ran'}_Gk') =|> State': = 9
            ∧ request (B, U, player_bank_prize, Prize')
end role
```

Figure 7. HLPSL for player and bank

Session

```
role session (
        U, B: agent,
        Ku, Kb: public_key
        Hash: hash
)
def=
local SND, RCV: channel(dy)
const ran, sk, gk, bank_player_ran, player_bank_n1, bank_player_n2, player_bank_rn, player_bank_prize:
protocol_id
composition
player (U, B, Ku, Kb, Hash, SND, RCV) ∧ bank (B, U, Kb, Ku, Hash, SND, RCV)

end role
```

Environment

```
role environment () def=
const
    u, b, i: agent,
    ku, kb, ki: public_key
    fhash: hash,
    xor: function,
    ran, sk, gk, bank_player_ran, player_bank_n1, bank_player_n2, player_bank_rn, player_bank_prize:
protocol_id
intruder_knowledge = {u, b, i, ku, kb, ki, inv (ki), fhash}
composition
session (u, b, ku, kb, fhash) ∧ session (u, i, ku, ki, fhash) ∧ session (b, i, kb, ki, fhash)

end role
```

Figure 8. HLPSL for session and environment

Table 5. Cross reference

Role	Notation	Specification in HLPSL
Session	*ran*	bank_player_ran
	N_1, N_2	player_bank_n1, bank_player_n2
	R	player_bank_rn
	prize	player_bank_prize

```
goal
    weak_authentication_on player_bank_n1
    weak_authentication_on bank_player_n2
    weak_authentication_on bank_player_ran
    weak_authentication_on player_bank_prize
    weak_authentication_on player_bank_rn

    secrecy_of id, sk, gk
end goal
```

Figure 9. HLPSL for goal

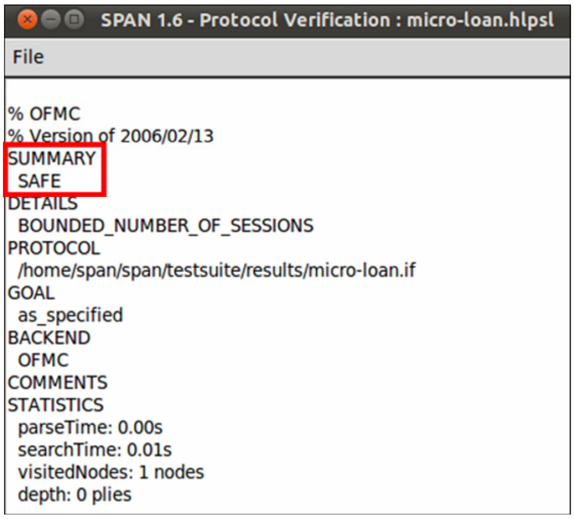


Figure 10. The effect of OFMC

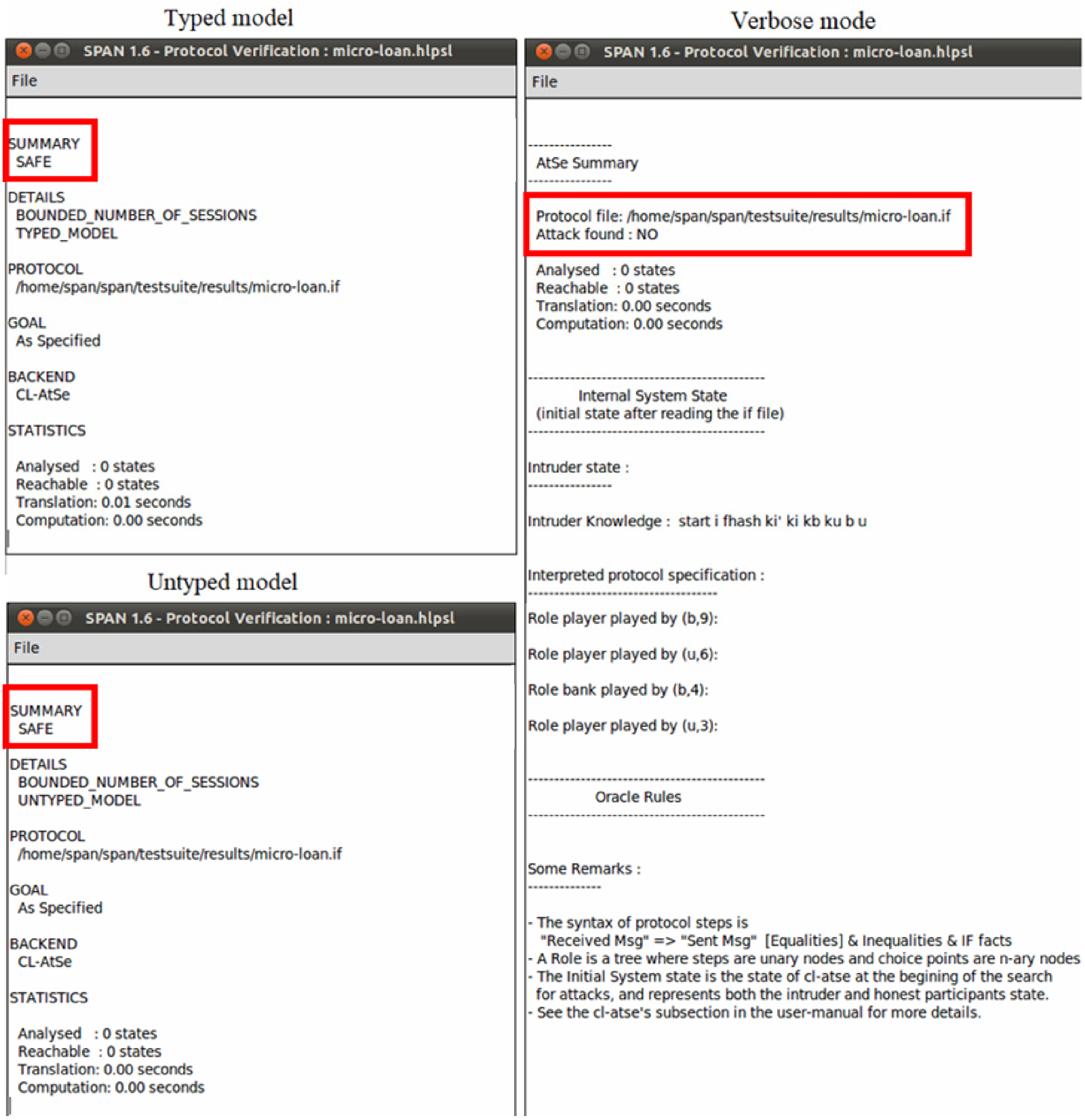


Figure 11. The effect of CL-ATSE

References

[1] J.-S. Lee, K.-S. Lin, A Robust e-commerce Service: Light-weight Secure Mail-order Mechanism, *Electronic Commerce Research and Applications*, Vol. 11, No. 4, pp. 388-396, August, 2012.

[2] J.-S. Lee, K.-S. Lin, An Innovative Electronic Group-buying System for Mobile Commerce, *Electronic Commerce Research and Applications*, Vol. 12, No. 1, pp. 1-13, January-February, 2013.

[3] J.-S. Lee, K.-J. Wei, Y.-C. Chen, Y.-H. Sun, Provable Secure Brand-new Multi-auction Mechanism with Dynamic Identity, *KSII Transactions on Internet and Information Systems*, Vol. 10, No. 12, pp. 6179-6205, December, 2016.

[4] H. Zhao, Q. Liu, G. Wang, Y. Ge, E. Chen, Portfolio Selections in P2P Lending: A Multi-objective Perspective, *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, USA, 2016, pp. 2075-2084.

[5] Y. Guo, W. Zhou, G. Luo, C. Liu, H. Xiong, Instance-based Credit Risk Assessment for Investment Decisions in P2P Lending, *European Journal of Operational Research*, Vol. 249, No. 2, pp. 417-426, March, 2016.

[6] X. Chen, B. Huang, D. Ye, The Role of Punctuation in P2P Lending: Evidence from China, *Economic Modelling*, Vol. 68, pp. 634-643, January, 2018.

[7] Z. Wang, C. Jiang, Y. Ding, X. Lyu, Y. Liu, A Novel Behavioral Scoring Model for Estimating Probability of Default over Time in Peer-to-peer Lending, *Electronic Commerce Research and Applications*, Vol. 27, pp. 74-82, January-February, 2018.

[8] C.-C. Chang, J.-S. Lee, Robust t-out-of-n Oblivious Transfer Mechanism Based on CRT, *Journal of Network and Computer Applications*, Vol. 32, No. 1, pp. 226-235, January, 2009.

[9] P. Kumar, A. Braeken, A. Gurtov, J. Iinatti, P. H. Ha, Anonymous Secure Framework in Connected Smart Home Environments, *IEEE Transactions on Information Forensics and Security*, Vol. 12, No. 4, pp. 968-979, April, 2017.

[10] D. Magazzeni, P. McBurney, W. Nash, Validation and Verification of Smart Contracts: A Research Agenda, *Computer*, Vol. 50, No. 9, pp. 50-57, September, 2017.

[11] J. P. Cruz, Y. Kaji, N. Yanai, RBAC-SC: Role-based Access Control Using Smart Contract, *IEEE Access*, Vol. 6, pp. 12240-12251, March, 2018.

[12] T. T. A. Dinh, R. Liu, M. Zhang, G. Chen, B. C. Ooi, J. Wang, Untangling Blockchain: A Data Processing View of Blockchain Systems, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 30, No. 7, pp. 1366-1385, July, 2018.

[13] J. Daemen, V. Rijmen, *The Design of Rijndael: AES – The Advanced Encryption Standard*, Springer-Verlag Berlin Heidelberg, 2002.

[14] A. Menezes, P. V. Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[15] L. Vigano, Automated Security Protocol Analysis with the AVISPA Tool, *Electronic Notes in Theoretical Computer Science*, Vol. 155, pp. 61-86, May, 2006.

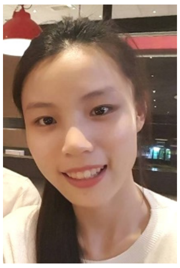
[16] D. Dolev, A. Yao, On the Security of Public Key Protocols, *IEEE Transactions on Information Theory*, Vol. 29, No. 2, pp. 198-208, March, 1983.

[17] M. Turuani, The CL-Atse Protocol Analyser, *International Conference on Rewriting Techniques and Applications*, Vol. 4098, Seattle, USA, 2006, pp. 277-286.

[18] D. Basin, S. Mödersheim, L. Viganò, OFMC: A Symbolic Model Checker for Security Protocols, *International Journal of Information Security*, Vol. 4, No. 3, pp. 181-208, June, 2005.

Biographies

Jung-San Lee received the B.S. degree in computer science and information engineering from National Chung Cheng University, Chiayi, Taiwan in 2002. He received his Ph.D. degree in computer science and information engineering in 2008 from National Chung Cheng University, Chiayi, Taiwan. Since 2017, he has worked as a professor in the Department of Information Engineering and Computer Science at Feng Chia University, Taichung, Taiwan. His current research interests include network management, electronic commerce, and blockchain.



Ying-Chin Chen received her M.S. degree in information engineering and computer science in Feng Chia University, Taichung, Taiwan in 2018. Her current research interests include information security, visual secret sharing, and blockchain.

Wei-Chih Hong received the B.S. and M.S. degrees in electrical engineering from NationalTaiwan University, Taipei, in 1996 and 1998, respectively. He received his Ph.D. degree at the Graduate Institute of Communication Engineering, National Taiwan University. During 1999-2003, he worked as an assistant researcher at the Telecommu-nication Laboratories of Chunghwa Telecom. Currently, he works as an assistant professor in the Department of Information Engineering and Computer Science at Feng Chia University, Taichung, Taiwan. His research interests are in the areas of wireless networks, information security, cryptanalysis, and blockchain.

Kun-Yin Lee is pursing his Industrial Ph.D. Program of Internet of Things in Feng Chia University, Taichung, Taiwan. His current research interests include information security and blockchain.

Ren-Kai Yang received his M.S. degree in information engineering and computer science in Feng Chia University, Taichung, Taiwan in 2019. His current research interests include network security and blockchain.