Information Sciences 477 (2019) 291–301

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Randomness invalidates criminal smart contracts

Yilei Wang[a,*], Andrea Bracciali[b], Tao Li[a], Fengyin Li[a], Xinchun Cui[a], Minghao Zhao[c]

[a] School of Information Science and Engineering, Qufu Normal University, China
[b] Computer Sciences and Mathematics School of Natural Sciences, Stirling University, United Kingdom
[c] School of Software, Tsinghua University, China

A R T I C L E I N F O

Article history:
Received 27 April 2018
Revised 3 October 2018
Accepted 28 October 2018
Available online 30 October 2018

Keywords:
Criminal smart contract
Randomness
Donation ratio

A B S T R A C T

A smart contract enforces specific performance on anonymous users without centralization. It facilitates payment equity in commerce by providing irreversible transactions. Smart contracts are also used for illegal activities such as money laundering and ransomware. Such contracts include criminal smart contracts (CSCs), proposed in CCS'16, that can be efficiently implemented in existing scripting languages. This aggravates concerns about the dangers of CSCs. However, *PublicLeaks*, a CSC for leaking private data, is conditionally implemented as it is influenced by various factors. For example, *PublicLeaks* does not necessarily reach a desirable terminal state for a criminal leaking private information, and other possible terminal states may invalidate the CSC. In this study, we propose a CSC based on *PublicLeaks* by formulating random factors such as the donation ratio. Our contract forks into five terminal states, including a unique one in *PublicLeaks* due to randomness. We simulated the maximal probabilities of these terminal states and found that the desirable terminal state in *PublicLeaks* is reachable with low probabilities (lower than 25%). The terminal state where the criminal fails to leak private information is attained with relatively high probabilities (over 65%). Therefore, our simulations show that CSCs are not always as powerful as expected, and the risk posed by them can be mitigated.

1. Introduction

A smart contract is "a set of promises, specified in digital form, including protocols within which the parties perform on these promises [39]. They may enforce specific activities such as addressing financial fraud [18,49], e-voting [27], bug bounty [7] and the blockchain-Internet of things (IoT) combination [11,29,31,44,45,50]. Moreover, they can be applied to cloud computing to enforce payments [9,10,41]. However, smart contracts may cause significant damage if they are targeted by criminals [5,22,36,42]. Smart contracts, although widely used, are far from perfect because of potential security issues [28,46,47]. For instance, in June 2016, the Decentralized Autonomous Organization (DAO) was attacked, resulting in the loss of approximately USD 60 million.

Smart contracts face two types of security issues: internal security concerns and external attacks (see Table 1). The former refers to security concerns within smart contracts and the latter, attacks implemented by smart contracts. Luu et al.

* Corresponding author.
E-mail addresses: wang_yilei2000@163.com (Y. Wang), abb@cs.stir.ac.uk (A. Bracciali), litao_ldu@163.com (T. Li), lfyin318@126.com (F. Li), cxcsd@126.com (X. Cui), mh-zhao17@mails.tsinghua.edu.cn (M. Zhao).

https://doi.org/10.1016/j.ins.2018.10.057
0020-0255/

Table 1
Security issues in smart contracts.

Internal security concerns		External attacks	
[3,33]	Vulnerabilities	[2]	CSCs
[4]	Correctness	[6,33]	Validation
[13]	Unknown vulnerabilities	[8]	Smart contracts in cybercrime
[23]	Privacy	[22]	CSCs
[30]	Robust	[43]	Destroy mining pools

proposed new security problems in smart contracts and enhanced their robustness [30]. Kosba et al. [23] proposed a decentralized system called Hawk that guarantees the privacy of smart contracts. Bhargavan et al. verified the runtime safety and correctness of smart contracts by translating them into F* [4]. Atzei et al. surveyed attacks launched using Ethereum smart contracts [3]. They discussed the problem of security vulnerabilities and provided a taxonomy of programming pitfalls. Dika proposed an updated taxonomy of all known vulnerabilities [13] and investigated security code analysis tools in Ethereum, including Oyente, Securify, and SmartCheck. Nikolić et al. [33] recently analyzed nearly one million contracts and reported that 34,200 of them were vulnerable; they implemented the MAIAN tool for concrete validation and manual analysis.

In addition to these internal security concerns, smart contracts are vulnerable to exploitation for illegal purposes. Velner et al. proposed an attack implemented by smart contracts in which the adversary can destroy mining pools [43]. Juels et al. [22] discussed criminal smart contracts (CSCs) that can be efficiently implemented on Ethereum and called for policy-related and technical safeguards for beneficial smart contracts [22]. Brunoni and Beaudet-Labrecque studied smart contracts in detail in the context of cybercrime [8]. Alharby and Moorse claimed that no solution has been proposed to solve the problems posed by CSCs [2]. This has increased concerns about smart contracts. No measure appears sufficient to prevent the threat posed by smart contracts, especially CSCs. Bigi et al. provided a formal method to verify the validation of smart contracts using game theory [6]. They analyzed the effects of uncertainty introduced by deposits on systems. Specifically, they used PRISM [16] to clarify the specific actions of protocols. Their work inspired discussions on the validation of smart contracts.

The main task in decentralized platforms is to enforce trust among people in the absence of a centralized entity [19–21]. Some specific problems become easier if the entities trust one another, such as through encryption schemes [15,48] and malware detection [40]. Motivated by [6,22], we revisit the validation of CSCs [22] in this study to consider a specific one, *PublicLeaks*, in which a dealer manages to illegally leak private information. Juels et al. claimed that *PublicLeaks* is efficiently implemented in Ethereum [22]. The terminal state here means the end state of a smart contract. *PublicLeaks* has a unique terminal state S_{succ} in which a dealer leaks private information or a secret after collecting sufficient donations. Other terminal states are available for a smart contract to leak a secret. For example, one terminal state is when the donation is insufficient to leak a given secret. Such terminal states derive from uncertain factors like the donation ratio. Furthermore, the dealer may cheat during the execution of a smart contract. All of these factors should be considered. The main contributions of this paper are as follows:

- We revisit the contract *PublicLeaks* [22] and follow the intuition that there is more than one terminal state in CSCs. This is because uncertain factors may bias contracts to a variety of end states. Therefore, *PublicLeaks* is conditionally established owing to random factors even though it can be efficiently implemented in Ethereum.
- We study several random factors that can influence the validity of *PublicLeaks* and accordingly propose a CSC called $PublicLeaks_{Random}$. This new contract has five terminal states, including S_{succ} of *PublicLeaks*.
- The maximum probabilities of each terminal state in $PublicLeaks_{Random}$ were simulated using PRISM. The results show that the maximum probability of reaching S_{succ} in $PublicLeaks_{Random}$ was not high (no more than 25%). Furthermore, the probability of terminal state S_{end} in which the dealer fails to leak secrets was relatively high (over 65%). Therefore, the dealer cannot implement CSCs, and the threat posed by them can be mitigated.

The remainder of this paper is organized as follows. Section 2 reviews the basic framework of *PublicLeaks* [22] and extends it to $PublicLeaks_{Random}$ by introducing randomness. Compared with the unique terminal state of *PublicLeaks*, $PublicLeaks_{Random}$ has five terminal states in case a secret is leaked. Section 3 analyzes the maximal probabilities of reaching each terminal state in $PublicLeaks_{Random}$. Simulation results show that the unique terminal state in *PublicLeaks* was reached with relatively low probability, thereby reducing the power of CSCs. Furthermore, the maximal probability of reaching the terminal state in which leaking fails was high. These results show that CSCs are not as powerful as expected, and their validity is undermined by randomness in the real world. Finally, Section 4 presents our conclusions as well as directions for future work in the study area.

2. Smart contract with randomness

2.1. Basic framework in [22]

In [22], the authors claimed that it is possible to leak a secret and collect donations by using smart contracts. We restate the basic workflow of *PublicLeaks* for the sake of clarity.

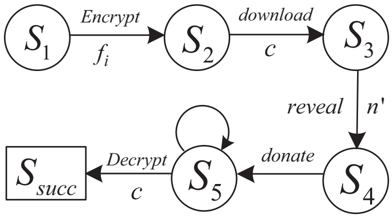

Fig. 1. State transitions of *PublicLeaks*.

Table 2
Random values for smart contract.

Parameter	Meaning
P_r	Probability of revealing correct secret keys corresponding to n'.
P_p	Ratio of k to cardinality of $\|Aud\|$.
P_d	Probability of dealer correctly decrypting the whole film.
P_l	Probability of dealer being amiable.

- S_1: The dealer divides film f (without copyright) into n segments f_i ($i \in [1, n]$), encrypts each segment f_i with secret key s_i, and sends them to the contract.
- S_2: The interested audience downloads all encrypted segments $c = \{c_i\}_{i\in[1,n]} = \{Enc_{k_i}(f_i)\}_{i\in[1,n]}$ from the contract.
- S_3: The contract selects subset $n' \subset [1, n]$ of n, and the dealer reveals the secret keys corresponding to n'.
- S_4: The audience donates money to the contract once it has successfully decrypted the segments using the revealed keys.
- S_5: The dealer decrypts all segments if he/she collects enough donations. Otherwise, he/she prefers to wait.
- S_{succ}: At the end of the contract, the dealer has collected enough money to sell the film, where the smart contract guarantees payment.

Fig. 1 shows six state transitions corresponding to the above steps. Let S_i denote the label of each state and S_{succ}, the terminal state in which the dealer successfully releases the film. Circle and rectangle nodes respectively denote nonterminal and terminal states; terminal states are reached once the dealer has collected a sufficient amount of donations.

2.2. Reconstruction of criminal smart contract considering randomness

Juels et al. focused on creating CSCs and left the conditions required to institute them as open problems [22]. For example, most people might wait for others to donate and decrypt a given film once the dealer releases all secret keys. In this case, no one donates if all users choose this strategy; this is similar to the famous prisoner's dilemma game [35]. It's a common problem in the real world, such as in free rider problems [14,17]. These open problems can be summarized as the following questions:

- What are the incentives for audiences to donate? It seems a better strategy for them to wait until others have donated enough money for private information to be leaked. Therefore, it is possible for most audiences to avoid donating. Juels et al. did not address this problem in detail [22].
- How much will each member of the audience donate and how many users will donate? Suppose each member donates the same amount in case he/she decides to donate. Let the donation ratio be the ratio of audience members who donate to the total number of audience members. Then, the donation ratio has a significant influence on the decryption of all film segments.
- When will the dealer have collected enough donations? In [22], the authors only mentioned that the dealer decrypts all segments when he/she has collected enough donations; however, they do not explicitly highlight the threshold of donations needed. This should be quantitatively stated so that the relationship between the donation ratio and total donations can be clearly defined.

To answer the above questions, some factors need to be introduced to smart contracts to influence the audiences choice of strategy. In this study, we formulate these factors as randomness in the smart contract. Note that the first three states are similar to those of *PublicLeaks* and the main forks start from state S_3. Randomness appears owing to several uncertain behaviors, including some probability definitions and decision conditions. A significant distinction from *PublicLeaks* is the introduction of a malicious or amiable dealer. A malicious dealer may sabotage the contract by deviating from it, such as by releasing incorrect secret keys. By contrast, an amiable dealer may honor the contract even if some conditions are not met, such as by releasing all secret keys even if enough donations are not collected. The random values and parameters with respect to dealers of different types are listed below and summarized in Tables 2 and 3.

Table 3
Parameters for smart contract.

Parameter	Meaning
d	Dealers deposit.
Aud	Set of all audience members.
Don	Subset of audience members who donate.
$\overline{Don}$	Subset of audience members who do not donate.
$vfilm$	Value of the whole film.
amt	Donation amount of each audience member.
k	Number of audience members who donate.
$donation$	Total donation amount.
$expected$	Dealers expected value. Herein, we set $expected = vfilm$.

- A malicious dealer may deliberately release n' incorrect secret keys with probability p_r.
- To prevent a malicious dealer from revealing incorrect secret keys, the dealer should first be required to deposit d to the contract. This deposit is not refunded if the dealer fails to reveal incorrect secret keys. We also assume that there is a small probability (e.g., 0.01) that the dealer refuse to deposit d to the contract.
- Let $Aud = \{Don, \overline{Don}\}$ denote the set of all audience members, where Don denotes audience members who donate and $\overline{Don}$, those who do not donate. We assume $Don \bigcup \overline{Don} = Aud$ for simplicity. Let $k = |Don|$ denote the cardinality of Don and $P_p = \frac{k}{|Aud|}$, the ratio of k to the cardinality of $|Aud|$.
- Let amt denote the amount donated by each audience member $i \in Don$ and $donation = k * amt$, the total donation.
- An honest dealer should release all n secret keys with probability P_d once $donation$ is greater than $vfilm$. A malicious dealer may release incorrect secret keys with probability $1 - P_d$.
- However, we allow an amiable dealer to decrypt the film with probability P_l by releasing all secret keys when the donations are close to his/her expected revenue. Let $P_l = 1 - \frac{|expected - donation|}{expected}$, where $donation$ denotes the total donations and $expected$, the dealer's expected revenue. We assume that $expected \geq vfilm$, where $vfilm$ is the value of the film. We also follow the idea in [6], where $vfilm$ and $Donation$ are exchanged between the dealer and Don once the smart contract is successfully implemented. In other words, the dealer sells the film at price $Donation$, and members of Don obtain $vfilm$ for the film. Furthermore, we assume that only Don may decrypt the entire film whereas $\overline{Don}$ may not. This assumption can effectively prevent $\overline{Don}$ from free-riding.

States of the $PublicLeaks_{Random}$ model, which is based on *PublicLeaks* [22], are explained in the following. Fig. 2 shows state transitions.

- S_{ini}: The smart contract is initiated. The dealer must submit a deposit d, and it is deducted if the dealer deviates from the smart contract.
- S_0: The dealer decides whether to deposit. The state changes to terminal state S_{abort} if he/she does not make the deposit. We assume that there is a small likelihood (e.g., 0.01) that the dealer does not make the deposit. Otherwise, the dealer deposit d and the state changes to S_1.
- S_{abort}: The contract is aborted. Note that the dealer deposits nothing.
- S_1: The contract collects deposit d. The dealer divides film f (without copyright) into n segments f_i ($i \in [1, n]$) and encrypts each segment f_i with secret key s_i. The state changes to S_2.
- S_2: Audience members belonging to Aud download all encrypted segments $c = \{c_i\}_{i\in[1,n]} = \{Enc_{k_i}(f_i)\}_{i\in[1,n]}$ from the contract. The state changes to S_3.
- S_3: The contract selects subset $n' \subset [1, n]$ of n. The state changes to S_4.
- S_4: The dealer reveals secret keys corresponding to n'. The state transitions to terminal state S_{fail} with probability $1 - P_r$ if the dealer fails to reveal the correct secret keys corresponding to n'. Otherwise, it proceeds to state S_5 with probability P_r.
- S_{fail}: The contract terminates and d is not refunded to the dealer.
- S_5: The set of audience members is divided into two subsets: Don and $\overline{Don}$. Audience members belonging to $\overline{Don}$ do not donate, and the state changes to S_6. Audience members belonging to Don donate amt, and the state changes to S_7.
- S_6: Audience members belonging to Don donate amt, and the state changes to S_8.
- S_7: Audience members belonging to $\overline{Don}$ do not donate anything, and the state changes to S_8.
- S_8: The contract collects donations from k members and sets $Donation = k * amt$. The state changes to S_9 if $Donation \leq vfilm$; otherwise, it changes to S_{10}.
- S_9: Normally, the dealer may not decrypt all segments as $Donation$ does not reach the expected value $expected$. The contract ends at state S_{end} and the deposit is refunded to the dealer. However, we allow an amiable dealer to release the entire film if $Donation$ is close to $expected$. Recall that the dealer decrypts the film by releasing all secret keys with probability P_l. Consequently, the state changes to S_{succ}.
- S_{10}: The dealer is willing to decrypt all segments, and the state changes to S_{succ}. However, there is still a small probability (e.g., $1 - P_d$) that a malicious dealer incorrectly decrypts files, and the state changes to S_{inc}.

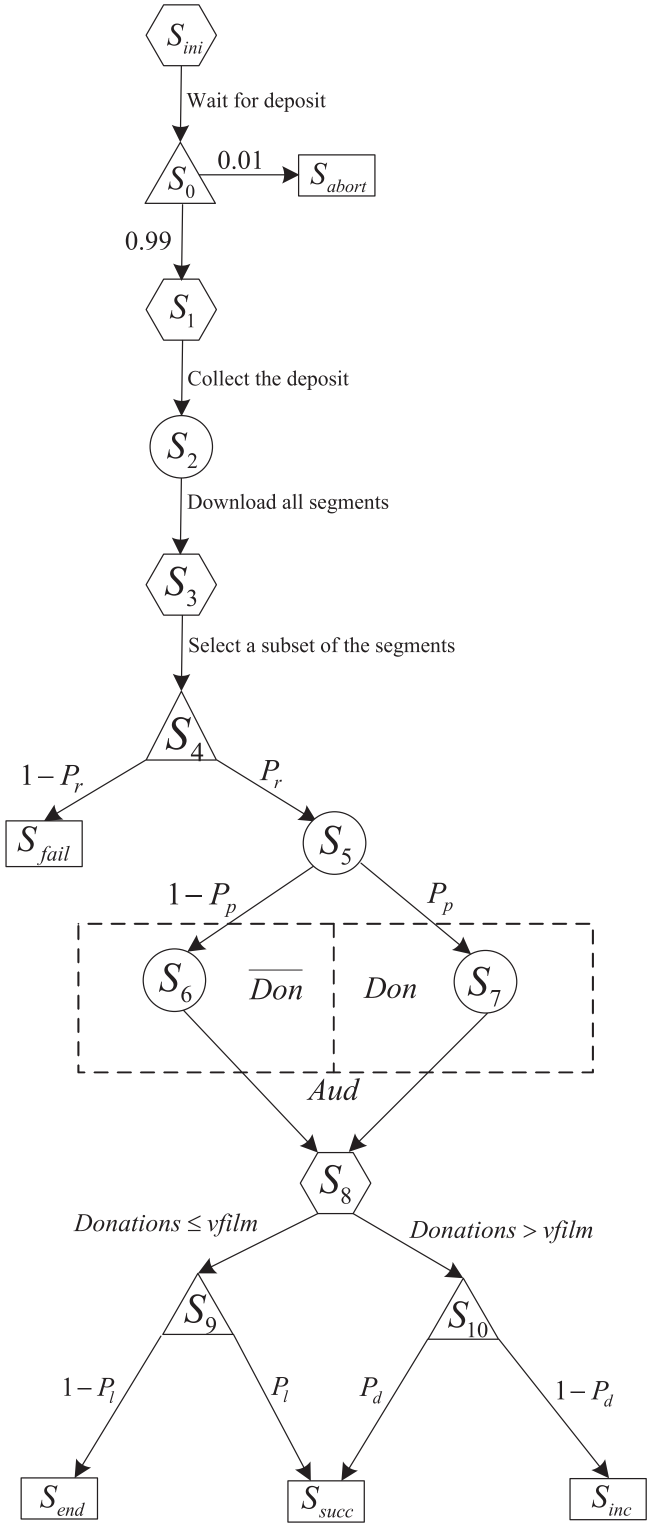

Fig. 2. State transitions of $PublicLeaks_{Random}$.

Table 4
Balance of $PublicLeaks_{Random}$.

State	Contract	Dealer	Audience members in *Don*	Audience members in $\overline{Don}$
S_{ini}, S_0, S_{abort}	0	*vfilm, d*	*amt*	*amt*
S_1, S_2, S_3, S_4, S_5, S_{fail}	*d*	*vfilm*	*amt*	*amt*
S_6, S_7, S_8, S_9, S_{10}	*d, Donation*	*vfilm*	0	*amt*
S_{end}	0	*vfilm, d*	*amt*	*amt*
S_{inc}	*d*	*vfilm*	*amt*	*amt*
S_{succ}	0	*d, Donation*	*vfilm*	*amt*

Table 5
Terminal states of $PublicLeaks_{Random}$.

States	Meaning
S_{abort}	Dealer refuses to deposit and the contract aborts.
S_{fail}	Dealer fail to decrypt part of the segments.
S_{end}	Contract ends normally, and the dealer does not decrypt all segments for audience members in *Don*.
S_{succ}	Contract is successfully executed, and the dealer decrypts all segments for audience members in *Don*.
S_{inc}	Dealer incorrectly decrypts all segments for audience members in *Don* but is detected.

- S_{end}: The contract is terminated. The donations are refunded to *Don* and deposit *d* is refunded to the dealer.
- S_{inc}: The contract is terminated. The donations are refunded to *Don* but deposit *d* is not refunded to the dealer.
- S_{succ}: The contract is terminated. The contract sends *Donation* and refunds the deposit to the dealer. Audience members, irrespective of whether they have donated, download the entire film.

The smart contract of $PublicLeaks_{Random}$ may solve the open problems mentioned above by setting $Donation > vfilm > amt$.

- The incentives for audiences to donate are similar to those in [22]. However, audiences in $\overline{Don}$ are not allowed to decrypt the film in $PublicLeaks_{Random}$. Therefore, audiences in $\overline{Don}$ cannot free-ride here.
- We consider the influence of the donation ratio on the decryption of all film segments by formulating the random value P_p. Furthermore, we discuss the influence of donation value *amt* on the decryption of the entire film.
- We consider different types of dealers—honest, malicious, and amiable; this results in a diversity of terminal states compared with those in [22]. We stress on the problems of when the dealer has collected enough donations and what he/she does once this has been done. For example, an amiable dealer may decrypt the entire film with probability P_l when the donations are close to but have not reached *vfilm*. By contrast, a malicious dealer may refuse to decrypt the entire film with probability $1 - P_d$ even if enough donations have been collected.

3. Simulations and results

Fig. 2 shows the state transitions of the smart contract, and Table 4 shows the balance of each participant. In Fig. 2, the rectangle denotes the terminal state, and circles, triangles, and hexagons denote different nonterminal states. Circles imply that the subject of the given state was part of the audience; the subject could belong to *Aud, Don*, or $\overline{Don}$. Triangles and hexagons imply that the subject of the given state was the dealer and the contract, respectively. The dashed rectangle denotes the set of audience members *Aud* divided into two subsets: *Don* and $\overline{Don}$. In Fig. 2, the terminal states of the contract are S_{abort}, S_{fail}, S_{end} ,S_{succ}, and S_{inc}. Table 5 lists the terminal states and their corresponding meanings.

In [22], the authors illustrated the probability of reaching state S_{succ} in smart contract *PublicLeaks*. In this study, we consider further possibilities for $PublicLeaks_{Random}$. For example, we consider the donation ratio P_p of audiences and the variation in donation *amt*. We also discuss different types of dealers, such as malicious and amiable. The former has a small probability $1 - P_d$ of incorrectly decrypting the entire film, whereas the latter has probability P_l of decrypting the entire film even if he/she does not collect enough donations. The main task is to learn the maximum probability of reaching each terminal state and the influence of randomness/parameters on it. However, it is challenging to prove through derivation of formulae. We thus simulated them by using PRISM [16,24], a useful probabilistic model checker that is widely used for modeling and verification of issues such as contract signing and analysis of anonymity [25,26,34,37,38]. PRISM has been evaluated in [12]. Therefore, one or more model properties are identified and implemented in PRISM's property specification language. The operator P is important in PRISMs property specification language; it can be used to calculate the probability of occurrence of an event, such as reaching a given state. For example, $Pmax = ?[F <= T\ target]$ denotes the maximum probability of reaching *target* within time *T*.

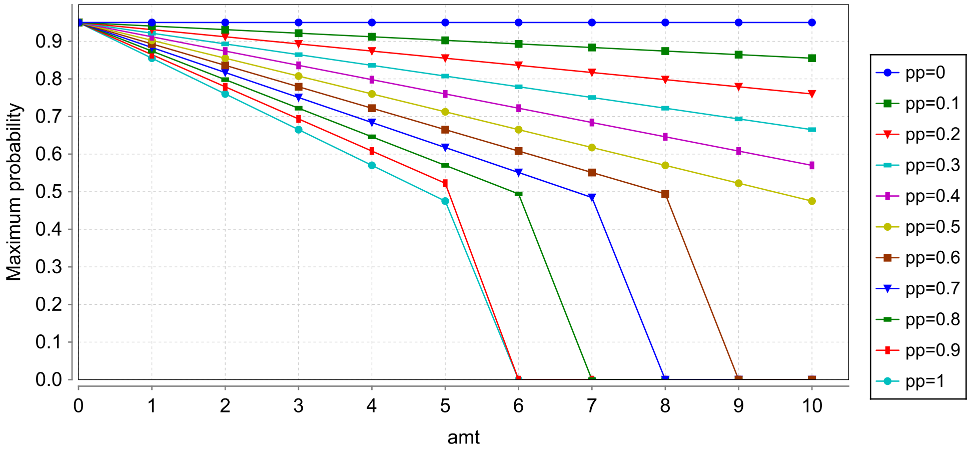

Fig. 3. Maximum probability of reaching S_{end}.

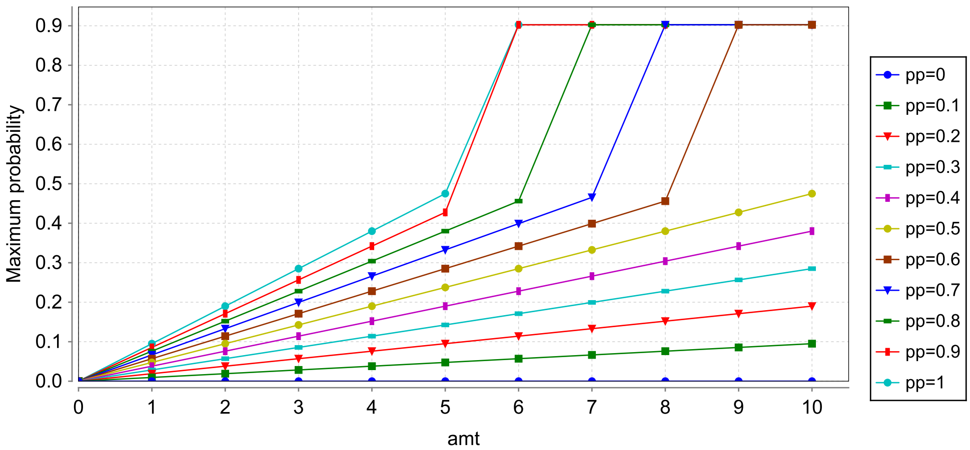

Fig. 4. Maximum probability of reaching S_{succ}.

In this study, we used the codes $Pmax = ?[F < 10\ s = end]$, $Pmax = ?[F < 10\ s = succ]$, and $Pmax = ?[F < 10\ s = inc]$[1] to denote the maximum probabilities of reaching states S_{end}, S_{succ}, and S_{inc}, within time T, respectively[2] Note that we did not analyze the maximum probabilities of termination states S_{acort} and S_{fail}; these were fixed at 0.01 and P_r, respectively. We only simulated the probabilities of terminal states S_{end}, S_{succ}, and S_{inc}.

We studied the combined influence of P_p and *amt* and the combined influence of P_p and P_l on the maximum probabilities of S_{end}, S_{succ}, and S_{inc}. Figs. 3, 4, and 5 show the simulation results of S_{end}, S_{succ}, and S_{inc}, respectively. The parameters were set as follows: $P_r = 0.95$, $p_d = 0.95$, $vfilm = 500$, $k = |Aud| * P_p$, $donate = k * amt$, and $P_l = 1 - \frac{|vfilm - donate|}{vfilm}$. In Fig. 3, the probability of reaching S_{end} was approximately 0.95 when the audience donated nothing. There was still a small probability, 0.05, of reaching S_{succ} as the dealer might have been an amiable one. Given the fixed value of *amt*, the higher the ratio of donations, the lower is the probability of reaching S_{end}. Given a fixed P_p, the higher the donation *amt*, the lower is the probability of reaching S_{end}. In other words, the probability of reaching S_{end} was inversely proportional to the donation ratio and value. The probability rapidly decreased to zero when the donation ratio P_p was higher than the threshold, 0.5. $P_p = .5$ was a watershed for the probability of S_{end} to decrease to zero. The situation for the probability of reaching S_{succ} was opposite that of reaching S_{end}.

In Fig. 4, the probability of reaching S_{succ} is proportional to the donation ratio and value. Similarly, $P_p = .5$ was a watershed for the probability of S_{succ} to reach 0.9. Here, the threshold was not one as a malicious dealer might have incorrectly decrypted the film even if he/she had collected enough donations. In Fig. 5, the probability of reaching S_{inc} is close to 0.05 when P_p is greater than 0.5. As with the threshold of P_p, the threshold of the donation value *amt* influenced the probabilities of reaching the terminal states, which changed with P_p. For example, in Fig. 4, the thresholds are $amt = 6$ and $amt = 7$

[1] We used $Pmax = ?[F < 10\ s = 11]$, $Pmax = ?[F < 10\ s = 12]$, and $Pmax = ?[F < 10\ s = 13]$ in the simulation as PRISM allows the states to be a set of integers.

[2] The time bound T in $F < T$ does not influence the final simulation results. Therefore, we set it to 10.

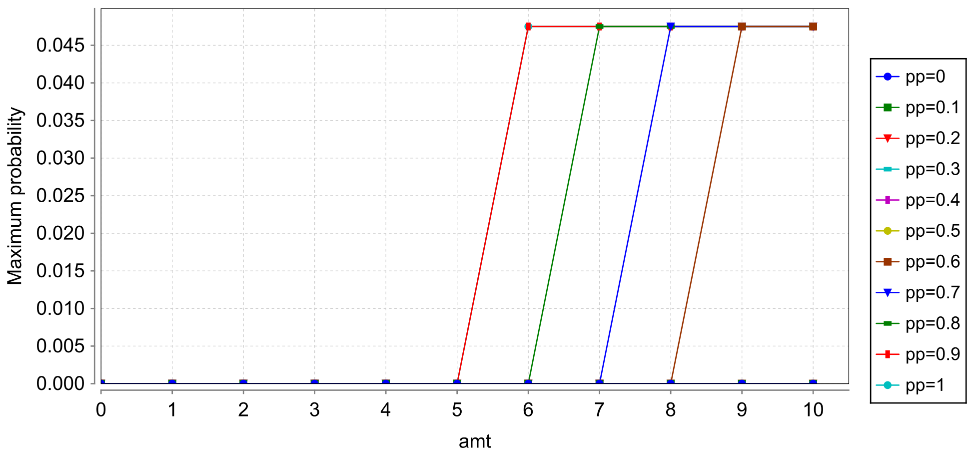

Fig. 5. Maximum probability of reaching S_{inc}.

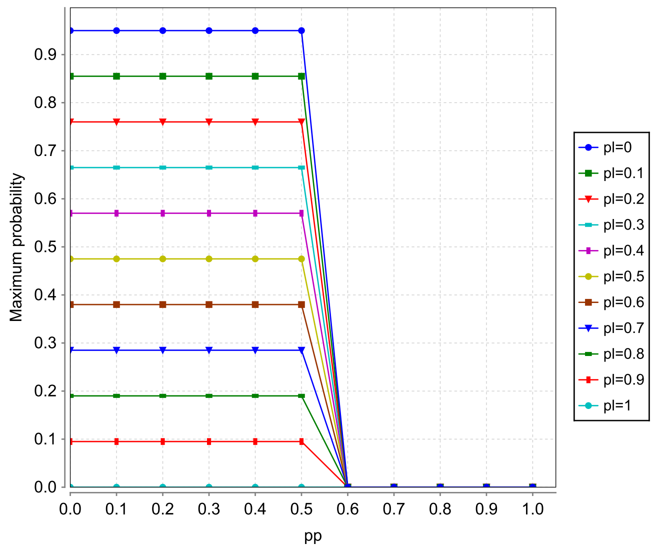

Fig. 6. Maximum probability of reaching S_{end} with fixed $P_r = P_d = 0.95$ and $amt = 1$.

when $P_p = .8$ and $P_p = .7$, respectively. In other words, the higher the P_p value, the smaller is *amt*. However, the thresholds were identical ($amt = 5$) when $P_p = .9$ and $P_p = 1$. This means that the threshold was not infinitely small. The smallest threshold was $amt = 5$ as $P_p = 1$ was the maximum probability. Figs. 3, 4, and 5 highlight the influence of these thresholds on the maximum probabilities. Therefore, the dealer should increase the thresholds of the donation ratio and value if he/she manages to improve the probability of reaching S_{succ}. However, the thresholds cannot be increased infinitely.

We studied the influence of P_p and P_l on the probabilities of reaching each terminal state. The parameters were set as follows: $P_r = 0.95$, $P_d = 0.95$, $amt = 1$, $vfilm = 500$, $k = |Aud| * P_p$, and $donate = k * amt$. Figs. 6, 7, and 8 show the simulation results. The probability of reaching S_{end} was inversely proportional to P_p and P_l, and that of reaching S_{succ} was proportional to P_p and P_l. However, the probability could not increase infinitely. In Fig. 7, the maximum probability decreases to 0.9 when $P_p = 1$. In Fig. 8, the probabilities were identical irrespective of P_l because they did not depend on P_l. Figs. 6, 7, and 8 show the thresholds. Unlike the thresholds in Figs. 3, 4, and 5, these thresholds were distinct for $P_p = .5$. The thresholds were watersheds indicating whether P_l influenced the maximum probabilities, especially in Figs. 6 and 7. In Fig. 7, the maximum probability of reaching S_{succ} is proportional to P_l. The highest probability is 0.95 when $P_p <= 0.5$ and $P_l = 1$. In other words, the smart contract could have been successfully executed with a higher probability, say 0.95, if the dealer had been amiable when P_p was higher than the threshold. However, P_l was ineffective for the maximum probability when $P_p < .5$. In Fig. 7, the maximum probabilities with different values of P_l reached a uniform value of 0.9 when $P_p > .5$. Therefore, it was better for the dealer to increase the donation ratio (e.g., higher than 0.5) if he/she had biased the smart contract to terminal state S_{succ}.

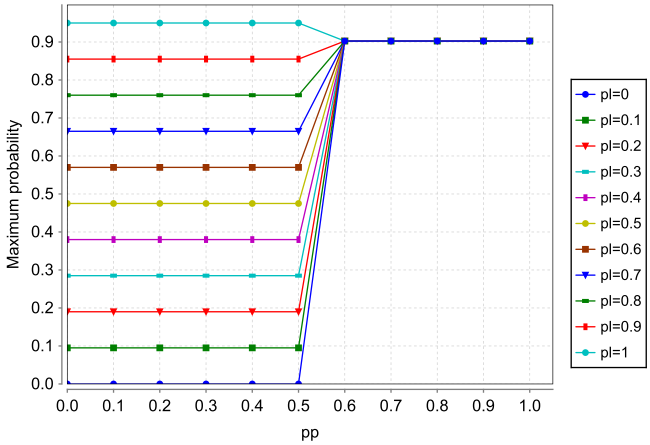

Fig. 7. Maximum probability of reaching S_{succ} where $P_r = P_d = 0.95$ and $amt = 1$.

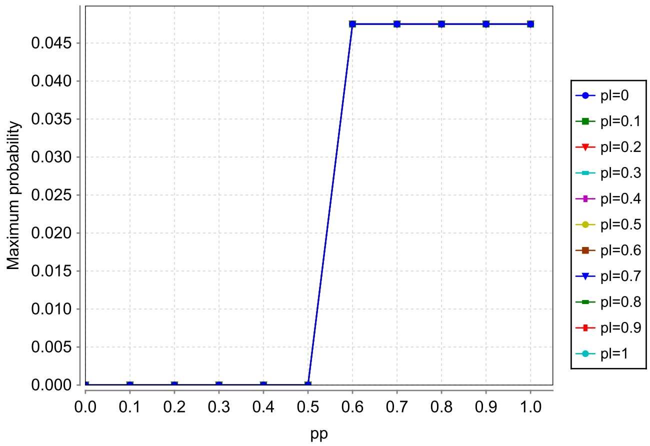

Fig. 8. Maximum probability of reaching S_{inc} where $P_r = P_d = 0.95$ and $amt = 1$.

The simulation results show that the donation ratio P_p and donation value *amt* had a significant influence on the success of the smart contract. The CSC could be executed with high probability (e.g., 0.9) if all audience members had donated (e.g., $P_p = 1$). This conclusion is consistent with that in [22]. The dealer may thus improve the probability of success of the CSC by manipulating P_p and *amt*. It is not particularly challenging for the dealer to increase *amt*. For example, the dealer can control *amt* by setting a minimum donation value. Therefore, the bottleneck is controlling the P_p value; it should be at least greater than 0.5 if the dealer manages to reach a high probability (e.g., 0.9). This means that the dealer should control at least half the audience, which is challenging. Donating behavior in smart contracts features free-riding [32], where people benefit without contribution. For example, in the popular P2P network Gnutella [1], approximately 70% of users do not contribute to the system. Therefore, we assumed that only 30% ($P_p = .3$) of audience members donated to CSCs. In Fig. 4, the probability is at most 0.25 when the CSC is successfully executed. In Fig. 7, the probability can be very high (e.g., 0.95) when $P_p = .3$ and $P_l = 1$. However, the premise is that the dealer is amiable. The P_l value depends on the P_p value according to its definition. Therefore, $P_l = 1$ is not set when $P_p = .3$.

The simulation results with respect to the maximum probability of reaching S_{succ} were not satisfactory, especially when P_p was low. It became challenging for the dealer to enforce the terms of the smart contract. In other words, although the CSC proposed by Juels et al. [22] is feasible in theory, it is challenging to implement.

4. Conclusions and future work

The property of payment enforcement in smart contracts is used by users to carry out illegal activities. As with real-world crimes, CSCs are not as powerful as assumed. In this study, we examined the validity of CSCs and found that some parameters may reduce their power. We biased the CSC using several forks by introducing random parameters.

We then proposed a CSC based on new parameters that has five terminal states, and we stress on three of them. The maximum probabilities of reaching each terminal state were simulated by using PRISM. The destructive power of our CSC was compromised as it was conditionally implemented with a relatively low probability. The power of CSCs diminishes with the introduction of randomness. Future work should focus on reducing the probability (below 0.3) of attaining the successful terminal state. Training smart contracts to escape illegal activities through machine learning is another interesting topic in this field.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 61502218 and 61771231), Natural Science Foundation of Shandong Province (ZR2017MF010 and ZR2017MF062), and Shandong Province Science and Technology Plan Projects (2015GSF116001). M. Zhao is funded by the National Natural Science Foundation of China under the grant No. 61602275, and the Open Project of Co-Innovation Center for Information Supply and Assurance Technology, Minghao Zhao under the grant No. ADXXBZ201702.

References

[1] E. Adar, B.A. Huberman, Free riding on gnutella, First Monday 5 (10) (2000) 2005.
[2] M. Alharby, A. van Moorsel, Blockchain-based smart contracts: a systematic mapping study, 2017, arXiv:1710.06372.
[3] N. Atzei, M. Bartoletti, T. Cimoli, A survey of attacks on ethereum smart contracts (SOK), in: Proceedings of the 2016 International Conference on Principles of Security and Trust, Springer, 2017, pp. 164–186.
[4] K. Bhargavan, A. Delignat-Lavaud, C. Fournet, A. Gollamudi, G. Gonthier, N. Kobeissi, N. Kulatova, A. Rastogi, T. Sibut-Pinote, N. Swamy, et al., Formal verification of smart contracts: Short paper, in: Proceedings of the ACM Workshop on Programming Languages and Analysis for Security, ACM, 2016, pp. 91–96.
[5] M.Z.A. Bhuiyan, J. Wu, G. Wang, Z. Chen, J. Chen, T. Wang, Quality-guaranteed event-sensitive data collection and monitoring in vibration sensor networks, IEEE Trans. Ind. Inf. 13 (2) (2017) 572–583.
[6] G. Bigi, A. Bracciali, G. Meacci, E. Tuosto, Validation of decentralised smart contracts through game theory and formal methods, in: Programming Languages with Applications to Biology and Security, 2015, pp. 142–161.
[7] L. Breidenbach, P. Daian, F. Tramr, A. Juels, Enter the hydra: towards principled bug bounties and exploit-resistant smart contracts, 2017, (Cryptology ePrint Archive, Report 2017/1090). https://eprint.iacr.org/2017/1090.
[8] L. Brunoni, O. Beaudet-Labrecque, Smart contracts and cybercrime: a game changer? Math. Struct. Model. 4 (44) (2017) 136–142.
[9] Z. Cai, H. Yan, P. Li, Z.A. Huang, C. Gao, Towards secure and flexible EHR sharing in mobile health cloud under static assumptions, Cluster Comput. 20 (3) (2017) 2415–2422.
[10] X. Chen, J. Li, X. Huang, J. Ma, W. Lou, New publicly verifiable databases with efficient updates, IEEE Trans. Dependable Secure Comput. 12 (5) (2015) 546–556.
[11] K. Christidis, M. Devetsikiotis, Blockchains and smart contracts for the internet of things, IEEE Access 4 (2016) 2292–2303.
[12] M.K. Dave Parker, Gethin Norman, Prism manual, 2017, (https://www.prismmodelchecker.org/manual/PropertySpecification/Introduction). [Online].
[13] A. Dika, Ethereum smart contracts: security vulnerabilities and security tools, NTNU, 2017 Master's thesis.
[14] M. Feldman, C. Papadimitriou, J. Chuang, I. Stoica, Free-riding and whitewashing in peer-to-peer systems, IEEE J. Sel. Areas Commun. 24 (5) (2006) 1010–1019.
[15] C. Gao, S. Lv, Y. Wei, Z. Wang, Z. Liu, X. Cheng, M-Sse: an effective searchable symmetric encryption with enhanced security for mobile devices, IEEE Access (2018), doi:10.1109/ACCESS.2018.2852329. Early publication online.
[16] A. Hinton, M. Kwiatkowska, G. Norman, D. Parker, Prism: A tool for automatic verification of probabilistic systems, in: Proceedings of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Springer, 2006, pp. 441–444.
[17] D. Hughes, G. Coulson, J. Walkerdine, Free riding on Gnutella revisited: the bell tolls? IEEE Distrib. Syst. Online 6 (6) (2005) 1–18.
[18] H. Hyvärinen, M. Risius, G. Friis, A blockchain-based approach towards overcoming financial fraud in public sector services, Bus. Inf. Syst. Eng. 59 (6) (2017) 441–456.
[19] R.H. Jhaveri, N.M. Patel, Y. Zhong, A.K. Sangaiah, Sensitivity analysis of an attack-pattern discovery based trusted routing scheme for mobile ad-hoc networks in industrial Iot, IEEE Access (2018), doi:10.1109/ACCESS.2018.2822945. Early publication online.
[20] L. Jiang, Y. Cheng, L. Yang, J. Li, H. Yan, X. Wang, A trust-based collaborative filtering algorithm for e-commerce recommendation system, J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-018-0928-7. Early publication online.
[21] W. Jiang, G. Wang, M.Z.A. Bhuiyan, J. Wu, Understanding graph-based trust evaluation in online social networks: methodologies and challenges, ACM Comput Surv 49 (1) (2016) 1–35.
[22] A. Juels, A. Kosba, E. Shi, The ring of Gyges: Investigating the future of criminal smart contracts, in: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, ACM, 2016, pp. 283–295.
[23] A. Kosba, A. Miller, E. Shi, Z. Wen, C. Papamanthou, Hawk: The blockchain model of cryptography and privacy-preserving smart contracts, in: Proceedings of the IEEE Symposium on Security and Privacy (SP), IEEE, 2016, pp. 839–858.
[24] M. Kwiatkowska, G. Norman, D. Parker, Prism 4.0: verification of probabilistic real-time systems, in: Proceedings of the International Conference on Computer Aided Verification, Springer, 2011, pp. 585–591.
[25] M. Kwiatkowska, G. Norman, D. Parker, Probabilistic model checking: advances and applications, in: Formal System Verification, 2018, pp. 73–121.
[26] G. Lenzini, S. Mauw, S. Ouchani, Security analysis of socio-technical physical systems, Comput. Electr. Eng. 47 (2015) 258–274.
[27] J. Li, Z. Liu, X. Chen, F. Xhafa, X. Tan, D.S. Wong, L-Encdb: a lightweight framework for privacy-preserving data queries in cloud computing, Knowl. Based Syst. 79 (2015) 18–26.
[28] T. Li, J. Li, Z. Liu, P. Li, C. Jia, Differentially private naive Bayes learning over multiple data sources, Inf. Sci. (Ny) (2018). Doi:10.1016/j.ins.2018.02.056. Early publication online.
[29] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, Y. Tang, An id-based linearly homomorphic signature scheme and its application in blockchain, IEEE Access 6 (1) (2018) 20632–20640.
[30] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, A. Hobor, Making smart contracts smarter, in: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, ACM, 2016, pp. 254–269.
[31] W. Meng, E.W. Tischhauser, Q. Wang, Y. Wang, J. Han, When intrusion detection meets blockchain technology: a review, IEEE Access 6 (99) (2018) 10179–10188.
[32] R. Mishra, Incentive schemes for mobile peer-to-peer systems and free riding problem: a survey, arXiv:1606.07785 (2016).
[33] I. Nikolić, A. Kolluri, I. Sergey, P. Saxena, A. Hobor, Finding the greedy, prodigal, and suicidal contracts at scale, arXiv:1802.06038 (2018).
[34] G. Norman, V. Shmatikov, Analysis of probabilistic contract signing, J. Comput. Secur. 14 (6) (2006) 561–589.
[35] M. Osborne, A. Rubinstein, A Course in Game Theory, MIT Press, 1994.
[36] K. Riad, L. Ke, Roughdroid: operative scheme for functional android malware detection, Secur. Commun. Netw. (2018), doi:10.1155/2018/80873032018. Early publication online.

[37] V. Shmatikov, Probabilistic analysis of anonymity, in: Proceedings of the 15th IEEE Computer Security Foundations Workshop (CSFW'02), 2002, pp. 119–128.
[38] V. Shmatikov, Probabilistic model checking of an anonymity system, J. Comput. Secur. 12 (3/4) (2004) 355–377.
[39] N. Szabo, Smart contracts: building blocks for digital markets, EXTROPY J. Trans. Thought(16) (1996).
[40] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, Z. Tian, Towards a comprehensive insight into the eclipse attacks of Tor hidden services, IEEE Internet Things J. (2018), doi:10.1109/JIOT.2018.2846624. Early publication online.
[41] H. Tian, Z. Chen, C.-C. Chang, Y. Huang, T. Wang, Z.-a. Huang, Y. Cai, Y. Chen, Public audit for operation behavior logs with error locating in cloud storage., Soft Comput. (2018), doi:10.1007/s00500-018-3038-8. Early publication online
[42] Z. Tian, Y. Cui, L. An, S. Su, X. Yin, L. Yin, X. Cui, A real-time correlation of host-level events in cyber range service for smart campus, IEEE Access (2018), doi:10.1109/ACCESS.2018.2846590. Early publication online.
[43] Y. Velner, J. Teutsch, L. Luu, Smart contracts make bitcoin mining pools vulnerable, in: Proceedings of the International Conference on Financial Cryptography and Data Security, Springer, 2017, pp. 298–316.
[44] C. Wang, J. Shen, Q. Liu, Y. Ren, T. Li, A novel security scheme based on instant encrypted transmission for internet-of-things, Security and Communication Networks (2018). DOI: 10.1155/2018/3680851. Early publication online,.
[45] C. Wang, J. Shen, Q. Liu, Y. Ren, T. Li, A novel security scheme based on instant encrypted transmission for internet-of-things, Security and Communication Networks (2018). DOI:10.1155/2018/3680851. Early publication online.
[46] Y. Wang, T. Li, H. Qin, J. Li, W. Gao, Z. Liu, Q. Xu, A brief survey on secure multi-party computing in the presence of rational parties, J. Ambient Intell. Human. Comput. 6 (6) (2015) 807–824.
[47] Z. Wu, L. Tian, P. Li, T. Wu, M. Jiang, C. Wu, Generating stable biometric keys for flexible cloud computing authentication using finger vein, Inf. Sci. (Ny) (2016) 431–447.
[48] L. Yang, Z. Han, Z. Huang, J. Ma, A remotely keyed file encryption scheme under mobile cloud computing, J. Netw. Comput. Appl. (106) (2018) 90–99.
[49] Y. Zhang, R.H. Deng, X. Liu, D. Zheng, Blockchain based efficient and robust fair payment for outsourcing services in cloud computing, Inf. Sci. (Ny) 462 (2018) 262–277.
[50] Y. Zhang, D. Zheng, R.H. Deng, Security and privacy in smart health: efficient policy-hiding attribute-based access control, IEEE Internet Things J. 5 (3) (2018) 2130–2145.