VDABSys: A Novel Security-Testing Framework for Blockchain Systems Based on Vulnerability detection

Jinfu Chen[1,2], Qiaowei Feng[1], Saihua Cai[1,2(✉)], Dengzhou Shi[1], Dave Towey[3], Yuhao Chen[1], and Dongjie Wang[1]

[1] School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China
caisaih@ujs.edu.cn

[2] Jiangsu Key Laboratory of Security Technology for Industrial Cyberspace, Jiangsu University, Zhenjiang 212013, China

[3] School of Computer Science, University of Nottingham Ningbo China, Ningbo 315100, China

Abstract. Blockchain technology is a popular solution for secure transactions in untrusted networks. However, with the growing number of blockchain applications, how to ensure the security of the blockchain system itself has become an urgent problem. In this paper, we propose a novel security-testing framework for blockchain systems based on a vulnerability-detection model. Our study involves an analysis and comparison with existing software-vulnerability analysis methods. Our framework first addresses each factor that impacts the security of the blockchain system, with a vulnerability attack graph being constructed using model-checking to describe the complete exploitation process of system vulnerabilities. Reliability Theory is used to quantitatively assess the vulnerability attack graph of the blockchain system, thereby providing a theoretical basis for evaluating its security. Finally, we verify the effectiveness and feasibility of the proposed security-testing framework for blockchain systems on an e-voting election blockchain system. The results from our extensive experiments show that our proposed method outperforms other formal-verification-based methods for detecting blockchain vulnerabilities, and also provides a scientific and reliable assessment of blockchain system security.

Keywords: Blockchain system · Vulnerability detection model · Formal theory · Vulnerability attack graph · Reliability theory

This work was partly supported by the National Key R&D Program of China (Grant no. 2020YFB1005501), the National Natural Science Foundation of China (NSFC) (Grant nos. 62172194, 62202206 and U1836116), the Natural Science Foundation of Jiangsu Province (Grant no. BK20220515), the China Postdoctoral Science Foundation (Grant no. 2023T160275), the Leading-edge Technology Program of Jiangsu Natural Science Foundation (Grant no. BK20202001), and the Qinglan Project of Jiangsu Province.

H. Duan et al. (Eds.): SecureComm 2023, LNICST 567, pp. 287–305, 2025.
https://doi.org/10.1007/978-3-031-64948-6_15

1 Introduction

Since its appearance with Bitcoin in 2009, blockchain technology has become one of the most prominent emerging technologies in recent years, and it has been widely applied in various social and economic scenarios [13,14,43]. Blockchain is a novel cryptographic distributed network-transaction bookkeeping system that offers support for decentralization, anonymity and tamper-evident features [29]. It may be considered as a decentralized database [20] that offers potential economic value. However, the complexity of blockchain systems makes it easy for them to be attacked, and the blockchain industry saw 427 security incidents in 2022, with estimated losses reaching as high as $3.52 billion [26], a rise of approximately 37.3% from the previous year. Therefore, it is crucial to ensure the security of blockchain systems to address the emerging security challenges.

Blockchain security technology plays a crucial role in ensuring the security and trustworthiness of blockchain systems. It encompasses several critical aspects, including cryptography, smart-contract security, consensus-mechanism security, distributed authentication technology, privacy protection, and prevention of DDoS attacks [28]. Cryptography is fundamental to ensure the data encryption and digital signature [50], while smart-contract security is an essential means of ensuring the contract logic correctness and security [35]. Consensus-mechanism security is critical to maintain the consistency and security of the entire blockchain system [46]. Additionally, distributed authentication technology [2], privacy protection [51], and prevention of DDoS attacks [25] are other important technologies that can enhance the security of blockchain systems. However, existing approaches often focus on improving specific aspects of blockchain systems, and may lack unified security testing and evaluation protocols for entire blockchain systems: This creates challenges for verifying the overall security and trustworthiness of blockchain systems. As a result, it is necessary to continuously explore and develop an efficient and reliable blockchain security framework to ensure the security of blockchain systems.

In this paper, we present a security-testing framework called VDABSys (Vuln erability Detection Approach for Blockchain Systems) that builds on the work in VDMBS (Vulnerability Detection Model for Blockchain System) [8]. The VDABSys utilizes the MAL (Model Action Logic) [19] to characterize the functional modules of blockchain systems, along with the CTL (Computational Temporal Logic) [39] language to specify the security requirements that the blockchain system must fulfill. These inputs are then used by the formal verification tool called IVY Workbench [6] to generate a vulnerability attack graph for the blockchain systems, describing and analyzing the system vulnerability using Reliability Theory [11]. With the proposed framework, the vulnerability of the blockchain system is quantitatively assessed to provide a theoretical basis for evaluating its security. The main contributions of this paper are as follows:

- We propose a novel security-testing framework called VDABSys to provide a theoretical foundation for evaluating the security of blockchain systems. The VDABSys framework first constructs a vulnerability-detection model for the

blockchain system, and then constructs a vulnerability attack graph based on the vulnerability detection model. Finally, VDABSys quantitatively evaluates the vulnerability attack graph to analyze the security of entire blockchain system.
- We propose a vulnerability attack graph construction-method based on model checking and state-transition diagrams, thereby providing insights into the possible attack paths and helping to identifying the vulnerabilities in blockchain systems. The proposed method generates the counter-examples using model-checking techniques to create the state-transition diagrams.
- We use Reliability Theory to quantify the vulnerability attack graph and select the appropriate model to calculate the corresponding system reliability function according to the applicable scenarios, thereby providing a theoretical basis for evaluating the security of the blockchain system.
- We perform an example analysis on a blockchain e-voting system to verify the feasibility of VDABSys, and then demonstrate and validate the effectiveness of the proposed VDABSys through experiments and comparisons with other techniques.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 describes our proposed approach. Section 4 presents a case study. Section 5 presents the experimental results and analysis. Finally, we conclude the paper in Sect. 6.

2 Related Work

The architecture of a blockchain can be broadly categorized into application layer, contract layer, network layer, consensus layer and data layer. A majority of research into blockchain vulnerability detection has concentrated on these layers [5,9,15].

Conventional testing methods, such as black box [1] and penetration testing [4], have primarily been utilized for assessing the application layer. However, these methods tend to focus only on the functionality or interfaces to the blockchain system, potentially ignoring interactions between multiple users, and other things. This could result in some vulnerabilities not being detected. They may also involve generation of large amounts of test cases, which can involve significant manual effort and lengthy detection times. Consequently, these traditional testing approaches can be inefficient. Most blockchain security incidents result from exploiting security vulnerabilities in smart contracts [42,47]. Accordingly, there has been significant research into the examination of smart contracts. This has included utilizing techniques such as fuzzing [23] and symbolic execution [3]. However, these methods have relatively simple detection strategies and may not ensure accurate detection, even if the same vulnerability occurs in different code segments. Moreover, the computation time required for symbolic execution of the entire path is prohibitively high [17] and the limitation of execution path depth could result in undetected vulnerabilities. Intrusion detection systems (IDSs) have typically been employed to secure the blockchain network

layer [36]. However, traditional IDSs may not detect attacks on smart contracts, consensus mechanism attacks, 51% attacks and other new types of threats specific to blockchain technology. The testing methods for consensus layer and data layer also often require extensive manual intervention and may exhibit high false alarm rates [16,44].

Although many studies have been conducted on blockchain security, most of them have focused on the specific levels or components of the blockchain system: There is lack of unified frameworks that can ensure the overall security of blockchain system.

Currently, different architectures and models of system security can be used as a guideline to build a framework for blockchain security. For example, a security architecture that considers the security requirements of the Product Lifecycle Information Management (PLIM) phase was proposed for the Internet of Things (IoT). This approach identified the most important components that can be included for security management of IoT, and analyzed the potential security threats [48]. In addition, a scalable and automatic approach for cybersecurity was proposed based on the concepts of isolation, interaction and representation: This framework could provide a structured and potentially automated security process to achieve integral and comprehensive security solutions [41]. The Security Reference Architecture combined high fidelity asset visibility and deep endpoint intelligence with business contexts to automate security and IT operations [32]. The Center for Internet Security (CIS) assists organizations in concentrating their security resources and expertise on protecting against cyber-attacks [10]. The Cybersecurity Framework suggested by the National Institute of Standards & Technology (NIST) offers a security framework to empower enterprises and enhance their ability to defend against cyber-attacks [34].

In order to overcome the difficulties mentioned above, we present a novel vulnerability-detection model for blockchain systems based on earlier work by Chen et al. [8]. The proposed model considers various factors that can impact the security of blockchain systems. To analyze the vulnerability of blockchain systems, this paper constructs the vulnerability attack graph using a model-checking method [12], that comprehensively portrays the exploitation process. It applies Reliability Theory [11] to quantitatively evaluate and analyze the vulnerabilities, thereby providing a theoretical foundation for enhancing the security of blockchain systems.

3 Security-Testing Framework for Blockchain Systems

In this paper, we propose a security-testing framework called VDABSys (Vulnerability Detection Approach for Blockchain Systems), which consists of three parts: system modeling, formal verification and security assessment. The VDABSys framework is illustrated in Fig. 1.

The detailed VDABSys steps are as follows:

Step 1: Based on the established vulnerability-detection model, get the set of system functions, and perform vulnerability detection on each. This approach

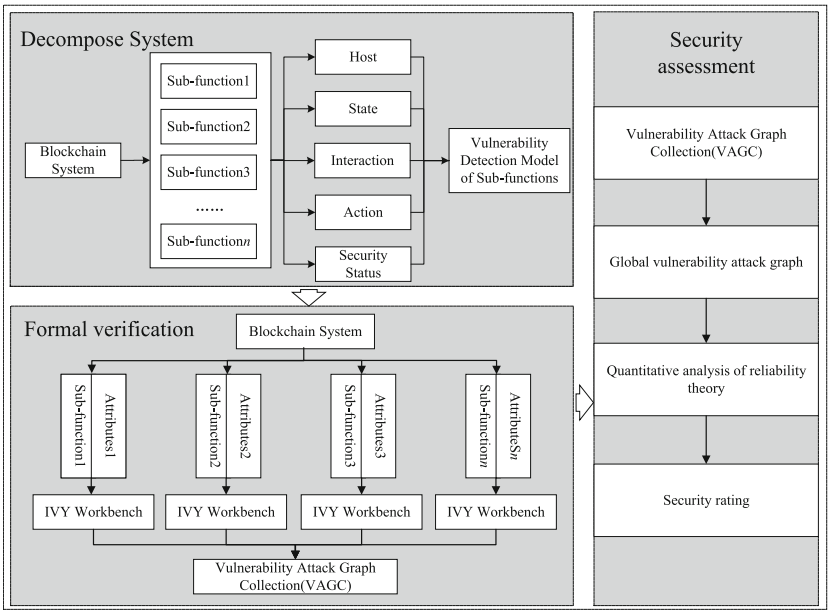

Fig. 1. Blockchain-system security-testing framework

breaks down the complex blockchain system into smaller sub-functions, making it easier to test each one individually, and reducing the overall testing difficulty.
Step 2: Formal methods are used to test each sub-function of the blockchain system. Firstly, the MAL (Model Action Logic) language is applied to formalize the variable definitions, state migration and migration conditions. Secondly, the CTL (Computing Tree Logic) language is used to extract and describe the set of security statutes that need to be satisfied by the current function. The MAL and CTL are then fed into the IVY Workbench formal verification toolset. If the verified property does not satisfy the condition in the currently tested sub-function (i.e., a vulnerability exists in this security statute), then the analyzer function of the workbench is selected to generate a counter-example for attacking this vulnerability. The counter-example is represented in the form of state-migration diagrams, which are used to generate the vulnerability attack graph of the sub-function. After testing all sub-functions, a collection of vulnerability attack graphs is produced. This process breaks down a complex system into individual sub-functions, which can reduce the testing difficulty and allow more thorough vulnerability detection.
Step 3: When all the vulnerability attack graphs for the sub-functions are generated, they are integrated to create a global vulnerability attack graph using unified integration processing for the whole blockchain system. Reliability theory is then applied to analyze and quantitatively assess the vulnerabilities, thereby providing a theoretical basis for evaluating the security of the blockchain system.

3.1 Decomposition of Blockchain System

In this section, we break down the complex blockchain system into smaller sub-functional modules to reduce the difficulty of performing security testing on the blockchain system based on the vulnerability-detection model. During decomposition, we formally define each functional module in terms of the seven dimensions considered in the model: function modules; hosts; interaction relations; intruder privileges; states; behaviors; and security protection objectives. We then perform vulnerability detection. We provide a brief overview of the proposed vulnerability-detection model for blockchain systems in Table 1. A comprehensive description of the model-construction process and its effectiveness in detecting vulnerabilities can be found in the work by Chen et al. [8].

Table 1. Overview of the vulnerability-detection model

Dimensions	Description
Function Modules	A blockchain system consists of a set of services or functions, where the function points (denoted f) are the basic units for the construction of blockchain security tests
Hosts	It includes the identifier of the host, the type of user using it, the services running on it, the software installed on it, and the set of vulnerabilities identified on it
Interaction Relation	The relation can be described by a triple relationship: $R \subseteq Host \times Host \times A$. For example: $R(h_i, h_j, b)$ means that host h_i has the ability to engage with host h_j through behavior b
Intruder Privilege	The level of permissions satisfying the full order relationship: None $<$ User $<$ Root
State	The five aspects of states of the blockchain system are expressed in terms of their number, name, the subject condition, the object condition, and the environment condition
Behaviors	The three aspects of behaviors that make a change in the state of vulnerability are expressed in terms of the source state, the target state, and the cost parameter ε
Security Protection Objective	The two aspects of the security protection objective that the blockchain system needs to satisfy during its operation are expressed in terms of the number and the detailed description of this security statute. The security attribute specification is a formal description of the safety attribute using branching time logic.

3.2 Construction of the Vulnerability Attack Graph

To construct a vulnerability attack graph, we formalize the system using the MAL language as well as extract and describe the security statute AGF using the CTL language. We use the IVY Workbench formal-verification toolset to

verify whether or not the AGF properties hold. If the AGF is false, the workbench analyzer function is used to generate a counter-example, which represents a path to an unsafe state. The counter-example is then transformed into a state-migration diagram, which is used to generate the vulnerability attack graph for the blockchain system. The construction of the vulnerability attack graph provides insights into the possible attack paths, and helps to identify the vulnerabilities in the blockchain systems.

Step 1: Extract the set of states S for the current test sub-function from the vulnerability-detection model. The initial state S_0 belongs to S, and the behavior ensemble A is a subset of $S \times S$, where a change in state from s to s' is denoted by the tuple (s, s').

Step 2: Establish the set of behaviors A for migration from each state to the next state, and then determine the set of security statutes $G = AGF$ that need to be satisfied by this sub-function according to the vulnerability model.

Step 3: Complete the formalization of variable definitions, state migrations and behavior descriptions using MAL language and represent current sub-function in a formalized manner. And then, describe the set of security statutes (G) using CTL language.

Step 4: Enter the information obtained from the previous steps into the IVY Workbench and then use the NuSMV model checker to iteratively search for the set S_{unsafe} of all unsafe states that do not conform to the security statute g as well as the set $R^g = R \cap (S_{unsafe} \times S_{unsafe})$ of all state transitions leading to the unsafe states. These sets can be extracted using the Logical States output of the TracesAnalyzer module in the IVY Workbench.

Step 5: Generate the sub-functional vulnerability attack graph $V_g = (S_{unsafe}, R^g, C, S_0^g, S_s^g, x)$, where S_{unsafe} is the set of unsafe state nodes, R^g is the set of edges, C is the set of conditions (including initial conditions, preconditions and postconditions), S_0^g is the set of unsafe initial states, $S_s^g = \{s \mid s \in S_{unsafe}\}$ is the set of insecure states that violate the security statute g, x can be any other information that needs to be referenced(such as open ports, IDS, etc.).

Step 6: Use the graphical tool Graphviz to integrate the set of sub-functional vulnerability attack graphs into the blockchain system vulnerability attack graph. This final (system-level) attack graph is output as a state-migration graph [27].

3.3 Quantitative Analysis of Risk Based on Vulnerability Attack Graphs

In this paper, we use Reliability Theory [11] to quantify the vulnerability attack graph. It identifies the reliability parameters used to describe the reliability of blockchain systems under evaluation. The main reliability parameters in the vulnerability attack graph of the blockchain systems are ε, $Re(c)$, and $E(c)$.

– ε: The success probability ε of an atomic behavior represents its cost, which is determined by the factors such as computational resources, human resources,

knowledge level, storage space, and time required for the behavior to be executed. We estimate ε based on the type of vulnerability, considering the vulnerability model of blockchain systems established in the previous section. Specifically, we assume that an attacker exploiting the same vulnerability generates several atomic behaviors with the same probability of success.

– $Re(c)$: The reliability function $Re(c)$ of the blockchain systems measures the overall security of the systems. It indicates the probability that the systems will maintain security at a specified cost c under specified conditions. In the context of vulnerability attack graphs, $Re(c)$ represents the probability that the systems remain secure after a certain number of atomic behaviors have been executed by the attacker.
Larger values of $Re(c)$ indicate a higher level of security, while smaller values indicate a greater vulnerability to attacks.
– $E(c)$ denotes the average attack cost on the blockchain systems.

Based on the architectural characteristics of the blockchain system, we classify the vulnerability state diagram into three models: serial; parallel; and composite. Then, we select the appropriate model to calculate the corresponding system reliability function according to the applicable scenarios.

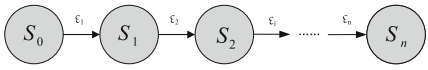

Fig. 2. Serial model

For the serial model shown in Fig. 2, if CO_i and ε_i are the cost and success probability of the atomic behavior, respectively, at step i, then $Re_s(c)$ is calculated using Equation (1):

$$\mathrm{Re}_s(c) = 1 - P\left(CO_1 + CO_2 + \ldots + CO_n \leq c\right) = \sum_{i=1}^{n} \frac{\prod_{j=1, j\neq i}^{n} \varepsilon_j e^{-\varepsilon_j c}}{\prod_{j=1, j\neq i} \left(\varepsilon_j - \varepsilon_i\right)} \quad (1)$$

Assumptions: $\forall i \neq j \rightarrow \varepsilon_i \neq \varepsilon_j; n \geq 2$

For the parallel model shown in Fig. 3, if CO_i and ε_i are again the cost and success probability of the atomic behavior at step i, then $Re_s(c)$ is calculated using Eq. (2):

$$\mathrm{Re}_s(c) = P\left(\min CO_i > c\right) = \prod_{i=1}^{n} P\left(CO_i > c\right) = e^{-\sum_{i=1}^{n} \varepsilon_i c} \quad (2)$$

For the composite model shown in Fig. 4, $Re_s(c)$ is calculated using a recursive approach, as in Eq. (3):

$$\mathrm{Re}_s(c) = \prod_{k \in in(s)} \left(\mathrm{Re}_k(c), \varepsilon_{k,s}\right) \quad (3)$$

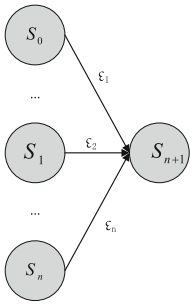

Fig. 3. Parallel model

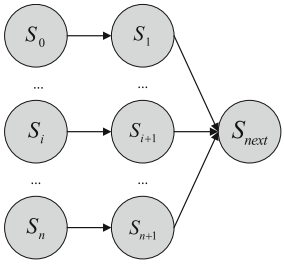

Fig. 4. Composite model

where s is the target node of the vulnerability attack graph; in(s) is the set of all input nodes to s; $\varepsilon_{k,s}$ is the probability of success of the atomic behavior corresponding to the directed edge from the input node k to the target node s; $Re_k(c)$ is the reliability function corresponding to input node k; and $(Re_k(c), \varepsilon_{k,s})$ is the reliability function corresponding to the path from the input node k to the target node s.

The vulnerability of the analyzed blockchain system should be calculated using a combination of the above three models. After obtaining the system $Re_s(c)$, the average cost of the system attack (written as $E(C)$), is calculated using Equation (4):

$$E(C) = \int_0^{+\infty} \mathrm{Re}_s(c)dc \tag{4}$$

Larger values of $E(C)$ indicate greater average costs for the attacker to complete the attack target, that is, the blockchain system is more secure. We map the results to the interval [0, 1), where larger values indicate more expensive attacks for the attacker, and thus a more secure blockchain system. The mapping function is given in Equation (5):

$$E'(C) \to \frac{\arctan E(C)}{\pi} + \frac{1}{2} \tag{5}$$

4 Case Study

In this section, we illustrate the feasibility and effectiveness of VDABSys using a simple e-voting blockchain system called BlockChainVoting [31] as an example.

4.1 The E-Voting Blockchain System

The e-voting blockchain system consists of three core modules: the election administrator module; the voting module for voters; and the voting management module. The main implemented functions are as follows:

(1) The election administrator module allows adding, deleting and updating of the candidates.
(2) The voting module allows voter registration, connection, verification and polling functions.
(3) The voting management module allows the creation, authorization, statistics and publishing functions of voting events.

4.2 The Process of Testing and Evaluation for the E-Voting Blockchain System

In this section, we report on the security testing and evaluation of BlockChainVoting with the proposed VDABSys system. First, the vulnerability-detection model is constructed for the system. Then, the vulnerability-detection model is verified using formal methods, and a vulnerability attack graph is generated based on the verification results. Finally, an expected cost of the attack is computed by choosing an appropriate reliability computation model based on the vulnerability attack graph, thereby assessing the security of the BlockChainVoting system.

Step 1: We construct the vulnerability-detection model for BlockChainVoting system in accordance with the approach introduced in Sect. 3.1. On the analysis of BlockChainVoting system, we construct the behaviors and state of three core functions (the election administrator module, the voting module and the voting management module). The set of voting module states is shown in Table 2 and a detailed description of each behavior is given in Table 3.

Table 2. The set of states of voting module

State ID	State name	Environment conditions	Object conditions	Subject conditions
S_0	Initial state	$\oslash$	Frequency = 0	CurNumber = 0
S_1	Vote	$RshTrust(full, user)$	Frequency = 1	CurNumber = 1
S_2	Pause	$RshTrust(full, manager)$	Frequency = 2	CurNumber = 1
S_3	Counting	$RshTrust(full, manager)$	Frequency = 1	CurNumber = 2

Table 3. The set of behaviors of the voting module

Behavior ID	Behavior name	Source state	Target sate	Cost parameter ε
a_1	Start	$\{S_0,S_2,S_3\}$	$\{S_1\}$	$\varepsilon_1 = 0.25$
a_2	Pause	$\{S_1\}$	$\{S_2,S_3\}$	$\varepsilon_2 = 0.25$
a_3	Add votes	$\{S_1\}$	$\{S_1\}$	$\varepsilon_3 = 0.5$
a_4	Count votes	$\{S_0,S_1,S_2\}$	$\{S_0,S_1\}$	$\varepsilon_4 = 0.5$
a_5	Cleared	$\{S_0,S_1,S_2,S_3\}$	$\{S_0\}$	$\varepsilon_5 = 0.5$

We also construct 15 security statutes and 17 CTL constraints: 14 of the CTL constraints are based on interactions in the BlockChainVoting system, and three are based on the intruder models for denial-of-service attacks, eclipse attacks and Sybil attacks. The set of statues that should meet the needs of the voting module is given in Table 4.

Table 4. The set of security statutes for the voting module

Statute ID	Description	CTL
g_1	The current number of votes cannot be less than 0	$AG(!(curNumber < 0))$
g_2	The current number of votes cannot be greater than the maximum number of voters	$AG(!(curStatus = Counting\&curNumber = Total))$
g_3	Each voting ID can only vote once	$AG(!(Frequency > 1))$
g_4	Voting is only possible during the required time period	$AG(!(curTime > begin\&curTime < end))$
g_5	The system can only perform polling operations when the polling status is turned on	$AG(!(curStatus = Pause\&IncNumber = true))$

Step 2: We use the MAL language to describe the variables, the migration of states and the behavior of the vulnerability detection model. We also use the CTL language to describe the security properties of the functional modules that must be checked. The MAL model and CTL properties are then fed into IVY Workbench for formal verification. Finally, we check the 17 CTL constraints, and find that ten security attribute statues violate the security statues of the BlockChainVoting system.

In the voting function module of the BlockChainVoting system, the five security properties are checked separately: g1 and g4 conformed to the security statute, but the remaining verification results were false. The logical state diagram of the counter-example is shown in Fig. 5.

We analyzed the three CTL constraints (g2, g3, and g5) that violated the security properties. With g2, the number of available votes was greater than the number of voters during the voting process, which suggests a system logic vulnerability of incorrectly limiting the times that each voter can vote. With g3, each voting ID could vote multiple times and did not have a threshold limit, which suggests that this system has a risk of DDoS attack. With g5, voters could continue to vote when voting was suspended, which indicates that this system has a logic vulnerability.

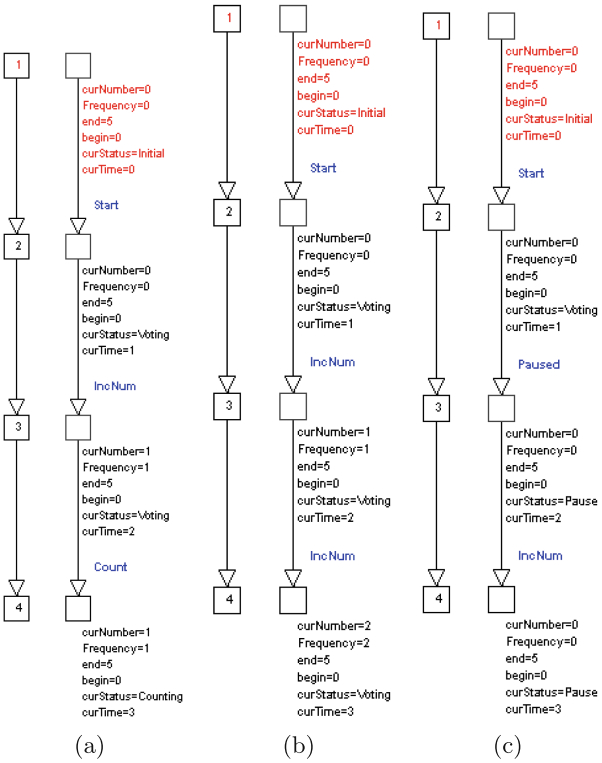

Fig. 5. Logical State Diagram of the counter-example

According to our proposed method, the corresponding vulnerability attack graph is constructed, as shown in Fig. 6.

Step 3: We analyze the above vulnerability attack diagram and compute the reliability function of BlockChainVoting system as:

$$\mathrm{Re}(c) = \sum_{i=1}^{n} \frac{\prod_{\substack{j=1 \\ j\neq i}}^{n} \varepsilon_j e^{-\varepsilon_j c}}{\prod_{\substack{j=1 \\ j\neq i}} (\varepsilon_j - \varepsilon_i)}$$

The corresponding average attack cost is $E(C) = \int_0^{+\infty} \mathrm{Re}_s(c)dc = \sum_{\mathrm{i}=1}^{4} \frac{1}{\varepsilon_i}$. Through the normalization process, the system's attack cost $E(C) = 0.2845$. This result indicates that the attacker only needs to pay a relatively small price to complete the attack and compromise the overall operation of the system. As a result, the overall security of this system is considered to be low.

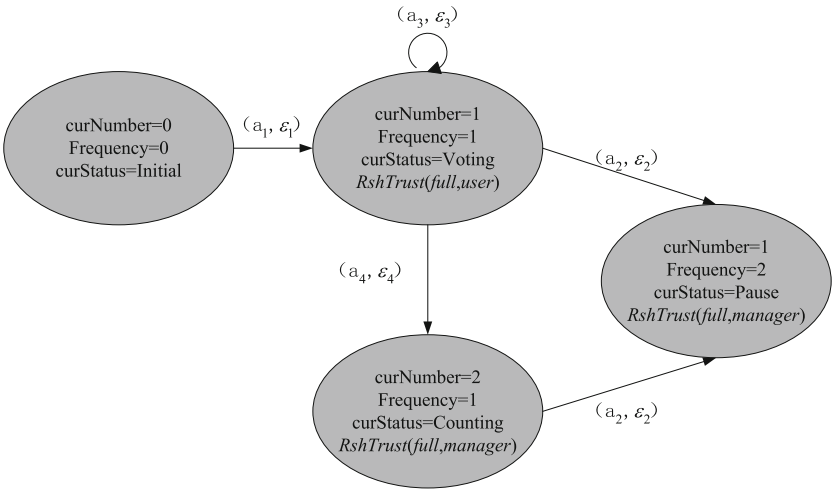

Fig. 6. The vulnerability attack graph of the BlockChainVoting system

5 Experimental Analysis

5.1 Experimental Setup

To verify the effectiveness of the proposed VDABSys, we conducted experiments with it and other blockchain vulnerability-detection approaches (the F* framework [18], ZEUS [24], and Coq [38]) for five different types of blockchain systems. All experiments were conducted on a computer with an i7-6700 CPU @ 2.90 GHz, 8 GB of RAM, and Windows 10 operating system. Here is a brief description of the five blockchain systems:

- BlockVotes [22]: An E-voting blockchain system, where administrators can improve operation of tasks for elections, and voters can vote in an election anytime and anywhere.
- Stellar Core [37]: A financial blockchain system that is a decentralized, federated peer-to-peer network that allows people to send payments anywhere in the world instantaneously, and with minimal fee.
- IPFS [21]: A blockchain system for storing files, designed to preserve and grow knowledge by making the web upgradeable, resilient, and more open.
- VeChain Thor [40]: A supply-tracking blockchain system that can record and track the production and supply chains of items to ensure their authenticity and compliance, preventing forgery and fraud.
- MedRec [33]: A medical-records management blockchain system designed to improve the security, reliability and interoperability of medical records. It uses blockchain technology to create a secure, accessible, decentralised medical-records storage system that makes it easier for parties such as doctors, patients and insurers to access and share medical records.

5.2 Experimental Result and Analysis

We first conducted vulnerability-detection modeling and formal verification for the five different types of blockchain systems, and then constructed the vulnerabi-lity-attack graph. Finally, we calculated the corresponding average attack cost using the vulnerability state diagram. The experimental results are shown in Table 5.

Table 5. VDABSys experimental results

System Name	System logic vulnerabilities	Smart Contract Vulnerabilities	Cyber-Attack Risks	Number of vulnerabilities	E(C)
BlockVotes	4	1	2	7	0.4523
Stellar Core	4	2	4	10	0.2699
IPFS	3	1	2	6	0.4702
VeChain Thor	5	2	4	11	0.2505
MedRec	3	2	3	8	0.3462

We grouped the detected vulnerabilities for the blockchain systems into three categories: system logic vulnerabilities; smart contract vulnerabilities; and cyber-attack risks. System logic vulnerabilities refer to errors in the design or implementation of the blockchain system that can lead to problems at the logical level (such as improper interaction between modules, insufficient input validation, improper exception handling, race condition and improper access control). Smart contract vulnerabilities are due to the unique characteristics of smart contracts, and include reentrancy vulnerabilities, insecure randomness generation vulnerabilities and time dependency vulnerabilities. Cyber-attack risks refer to the cyber-attacks faced by blockchain systems, such as DDoS attacks [25], eclipse attacks [45], Sybil attacks [49], timestamp attacks [7], and 51% attacks [30].

According to the experimental results shown in Table 5, VDMBS (the system modeling part of VDABSys) can detect several potential vulnerabilities through formal verification, with most vulnerabilities being related to system logic, followed by cyber-attack risks. The average cost of a system attack on the five blockchain systems calculated by VDABSys is much less than 1.0. Among the five blockchain systems, the average cost of Stellar Core and VeChain Thor are the lowest. The experimental results indicate that the proposed VDABSys is effective in detecting vulnerabilities in different types of blockchain systems.

We also analyzed and compared the proposed VDABSys with existing formal-verification-based blockchain vulnerability-detection approaches, the experimental results are shown in Table 6.

As shown in Table 6, the existing formal-verification-based blockchain vulnerability-detection approaches primarily focus on the security of the smart contract layer in blockchain systems without considering the overall state of the blockchain system during runtime. Compared with them, the proposed VDABSys examines the interaction security of smart contracts as well as the interactions and state migration of other modules in the blockchain systems. Therefore,

Table 6. Comparison of formal-verification-based blockchain vulnerability-detection approaches

The types of vulnerabilities		formal-verification-based blockchain vulnerability-detection approaches			
		VDABSys	F*	ZEUS	Coq
System logic vulnerabilities	BlockVotes	4	2	1	0
	Stellar Core	4	1	1	0
	IPFS	3	1	1	0
	VeChain	5	2	2	0
	MedRec	4	1	1	0
Smart contract vulnerabilities	BlockVotes	1	0	1	1
	Stellar Core	2	1	1	0
	IPFS	1	1	0	1
	VeChain	2	2	1	1
	MedRec	2	0	0	0
Cyber-attack risks	BlockVotes	2	0	0	0
	Stellar Core	4	0	0	0
	IPFS	2	0	0	0
	VeChain	4	0	0	0
	MedRec	3	0	0	0

compared to other methods that only focus on a specific part of the blockchain system, VDABSys is able to examine both the modules and the interactions between different modules in a comprehensive manner, thus enabling the discovery of potential vulnerabilities in different modules. As a result, VDABSys can detect more system logic vulnerabilities than the other approaches. It can also detect the cyber-attack risks, which the other approaches were not able to do. Our framework uses Reliability Theory and vulnerability attack graphs to enable a quantitative analysis of the blockchain system security; other methods are limited to only the vulnerability detection. Our proposed approach thus represents a reliability enhancement compared to existing formal methods approaches.

6 Conclusion

With the continuous development of computer technology, blockchain systems are very popular in network applications. Due to the complexity of these systems, there are significant security risks that must be addressed. Despite the development of various blockchain security technologies, there is not yet a comprehensive and unified approach, which represents a gap and vulnerability for blockchain security. To address these challenges, we present a new framework called VDABSys (Vulnerability Detection Approach for Blockchain Systems) for testing the security of blockchain systems.

This proposed VDABSys is based on the vulnerability-detection model VDMBS proposed by Chen et al. [8]. VDABSys constructs a vulnerability attack

graph using model checking and state transition diagrams, which can help to identify potential attack paths and vulnerabilities in blockchain systems. VDABSys applies Reliability Theory to quantify the vulnerability attack graph and selects an appropriate model to calculate the corresponding system reliability based on the specific scenarios, which can provide a rigorous foundation for evaluating the security of blockchain systems. Our experimental results confirm that the VDABSys can detect more system-logic vulnerabilities than other formal-verification-based blockchain vulnerability-detection approaches, and it can also detect the cyber-attack risks, which the other approaches were not able to do. Overall, our proposed VDABSys framework offers a novel and promising method for assessing the security of blockchain systems.

In the future, we would like to focus on refining the attack-success probability ε for each step in the vulnerability attack. We will also consider the factors such as computational resources, human resources, time and storage space to achieve more accurate quantitative vulnerability assessments, thereby further enhancing the effectiveness of the proposed framework and contributing to the overall improvement of blockchain system security.

References

1. Aggarwal, A., Shaikh, S., Hans, S., Haldar, S., Ananthanarayanan, R., Saha, D.: Testing framework for black-box AI models. In: 2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), pp. 81–84. IEEE (2021)
2. Babu, E.S., Dadi, A.K., Singh, K.K., Nayak, S.R., Bhoi, A.K., Singh, A.: A distributed identity-based authentication scheme for internet of things devices using permissioned blockchain system. Expert. Syst. **39**(10), e12941 (2022)
3. Badruddoja, S., Dantu, R., He, Y., Upadhayay, K., Thompson, M.: Making smart contracts smarter. In: 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), pp. 1–3. IEEE (2021)
4. Bhardwaj, A., Shah, S.B.H., Shankar, A., Alazab, M., Kumar, M., Gadekallu, T.R.: Penetration testing framework for smart contract Blockchain. Peer-to-Peer Netw. Appl. **14**, 2635–2650 (2021)
5. Cai, M., Sun, Z., Deng, X.: multidimensional observation of blockchain security. In: 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 1463–1466. IEEE (2021)
6. Campos, J.C., Sousa, M., Alves, M.C.B., Harrison, M.D.: Formal verification of a space system's user interface with the IVY workbench. IEEE Trans. Hum.-Mach. Syst. **46**(2), 303–316 (2015)
7. Chaganti, R., et al.: A comprehensive review of denial of service attacks in blockchain ecosystem and open challenges. IEEE Access **10**, 96538–96555 (2022)
8. Chen, J., Feng, Q., Cai, S., Shi, D., Rexford, N.A.S.: VDMBS: A novel vulnerability detection model for blockchain systems integrating multiple security factors based on formal technology.[in preparation]
9. Chepurnoy, A., Rathee, M.: Checking laws of the blockchain with property-based testing. In: 2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE), pp. 40–47. IEEE (2018)
10. CIS: CIS critical security controls. https://www.cisecurity.org/controls/ (2017)

11. Cui, T., Li, S.: System movement space and system mapping theory for reliability of IoT. Futur. Gener. Comput. Syst. **107**, 70–81 (2020)
12. Dang, S., Cao, S.: Regulation-as-a-service: model checking for decision-making behaviors in price-sensitive service systems. In: 2021 IEEE International Conference on Services Computing (SCC), pp. 1–12. IEEE (2021)
13. Delgado-Mohatar, O., Tolosana, R., Fierrez, J., Morales, A.: Blockchain in the internet of things: architectures and implementation. In: 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1072–1077. IEEE (2020)
14. Drăgulinescu, A.M., et al.: Smart watering system security technologies using blockchain. In: 2021 13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), pp. 1–4. IEEE (2021)
15. Feng, S., He, J., Cheng, M.X.: Security analysis of block withholding attacks in blockchain. In: ICC 2021-IEEE International Conference on Communications, pp. 1–6. IEEE (2021)
16. Gao, J., et al.: Towards automated testing of blockchain-based decentralized applications. In: 2019 IEEE/ACM 27th International Conference on Program Comprehension (ICPC), pp. 294–299. IEEE (2019)
17. Ghaleb, A., Pattabiraman, K.: How effective are smart contract analysis tools? Evaluating smart contract static analysis tools using bug injection. In: Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis, pp. 415–427. ACM (2020)
18. Grishchenko, I., Maffei, M., Schneidewind, C.: A semantic framework for the security analysis of Ethereum smart contracts. In: Bauer, L., Küsters, R. (eds.) POST 2018. LNCS, vol. 10804, pp. 243–269. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-89722-6_10
19. Harrison, M.D., Freitas, L., Drinnan, M., Campos, J.C., Masci, P., di Maria, C., Whitaker, M.: Formal techniques in the safety analysis of software components of a new dialysis machine. Sci. Comput. Program. **175**, 17–34 (2019)
20. Heo, H., Shin, S.: Behind block explorers: public blockchain measurement and security implication. In: 2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS), pp. 216–226. IEEE (2021)
21. IPFS: IPFS. https://github.com/ipfs/ipfs (2013)
22. Ivan: BlockVotes. https://github.com/yfgeek/BlockVotes (2018)
23. Jiang, B., Liu, Y., Chan, W.K.: ContractFuzzer: fuzzing smart contracts for vulnerability detection. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, pp. 259–269. ACM (2018)
24. Kalra, S., Goel, S., Dhawan, M., Sharma, S.: ZEUS: analyzing safety of smart contracts. In: Ndss, pp. 1–12. The Internet Society (2018)
25. Kumar, R., Kumar, P., Tripathi, R., Gupta, G.P., Garg, S., Hassan, M.M.: A distributed intrusion detection system to detect DDoS attacks in blockchain-enabled IoT network. J. Parall. Distrib. Comput. **164**, 55–68 (2022)
26. Lab, C.B.S.: Know chuangyu blockchain security lab: annual summary of blockchain security incidents in (2022). https://www.odaily.news/post/5184343 (2023)
27. Lee, S.M., Park, S., Park, Y.B.: State diagram based iot ecosystem's IoT device Conflict problem migration method. In: 2018 International Conference on Information Networking (ICOIN), pp. 684–688. IEEE (2018)
28. Leng, J., Zhou, M., Zhao, J.L., Huang, Y., Bian, Y.: Blockchain security: a survey of techniques and research directions. IEEE Trans. Serv. Comput. **15**(4), 2490–2510 (2020)

29. Lewis-Pye, A., Roughgarden, T.: How does blockchain security dictate blockchain implementation? In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, pp. 1006–1019 (2021)
30. Liu, Y., et al.: A blockchain-based decentralized, fair and authenticated information sharing scheme in zero trust internet-of-things. IEEE Trans. Comput. **72**(2), 501–512 (2022)
31. Mehta, A.: BlockChainVoting. https://github.com/mehtaAnsh/BlockChainVoting (2021)
32. Meltzer, D.: Security reference architecture: a practical guide to implementing foundational controls. https://www.tripwire.com/ (2017)
33. MITMediaLab: MedRec. https://github.com/mitmedialab/MedRec (2013)
34. NIST: Cybersecurity framework. https://www.nist.gov/cyberframework (2018)
35. Piantadosi, V., Rosa, G., Placella, D., Scalabrino, S., Oliveto, R.: Detecting functional and security-related issues in smart contracts: a systematic literature review. Softw. Pract. Experience **53**(2), 465–495 (2023)
36. Sohi, S.M., Seifert, J.P., Ganji, F.: RNNIDS: enhancing network intrusion detection systems through deep learning. Comput. Secur. **102**, 102151 (2021)
37. Stellar: Stellar-Core. https://github.com/stellar/stellar-core (2015)
38. Sun, T., Yu, W.: A formal verification framework for security issues of blockchain smart contracts. Electronics **9**(2), 255 (2020)
39. Tian, D., et al.: Two-phase motion planning under signal temporal logic specifications in partially unknown environments. IEEE Trans. Industr. Electron. **70**(7), 7113–7121 (2023)
40. VeChain: THOR. https://github.com/vechain/thor (2018)
41. Villalón-Fonseca, R.: The nature of security: a conceptual framework for integral-comprehensive modeling of IT security and cybersecurity. Comput. Secur. **120**, 102805 (2022)
42. Wan, Z., Xia, X., Lo, D., Chen, J., Luo, X., Yang, X.: Smart contract security: a practitioners' perspective. In: 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), pp. 1410–1422. IEEE (2021)
43. Wilczyński, A., Kołodziej, J., Grzonka, D.: Security aspects in blockchain-based scheduling in mobile multi-cloud computing. In: 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), pp. 696–703. IEEE (2021)
44. Wu, Z., et al.: Kaya: a testing framework for blockchain-based decentralized applications. In: 2020 IEEE International Conference on Software Maintenance and Evolution (ICSME), pp. 826–829. IEEE (2020)
45. Xu, G., Guo, B., Su, C., Zheng, X., Liang, K., Wong, D.S., Wang, H.: Am i eclipsed? a smart detector of eclipse attacks for Ethereum. Comput. Secur. **88**, 101604 (2020)
46. Yadav, A.K., Singh, K., Amin, A.H., Almutairi, L., Alsenani, T.R., Ahmadian, A.: A comparative study on consensus mechanism with security threats and future scopes: Blockchain. Comput. Commun. **201**, 102–115 (2023)
47. Yadav, J.S., Yadav, N.S., Sharma, A.K.: Security analysis of smart contract based rating and review systems: the perilous state of blockchain-based recommendation practices. Connect. Sci. **34**(1), 1273–1298 (2022)
48. Yousefnezhad, N., Malhi, A., Keyriläinen, T., Främling, K.: A comprehensive security architecture for information management throughout the lifecycle of IoT products. Sensors **23**(6), 3236 (2023)
49. Yu, B., Xu, C.Z., Xiao, B.: Detecting Sybil attacks in VANETs. J. Parall. Distrib. Comput. **73**(6), 746–756 (2013)

50. Zhang, B., Xu, J., Wang, X., Zhao, Z., Chen, S., Zhang, X.: Research on the construction of grain food multi-chain blockchain based on zero-knowledge proof. Foods **12**(8), 1600 (2023)
51. Zhang, M., Yang, M., Shen, G., Xia, Z., Wang, Y.: A verifiable and privacy-preserving cloud mining pool selection scheme in blockchain of things. Inf. Sci. **623**, 293–310 (2023)