Proceedings of CONF-MLA 2024 Workshop: Securing the Future: Empowering Cyber Defense with Machine Learning and Deep Learning
DOI: 10.54254/2755-2721/110/2024MELB0089

Securing the Chain: Comprehensive Analysis of Vulnerabilities and Defense Mechanisms in Blockchain Systems

Junxian Chen

Telecommunication, Beijing University of Post and Telecommunication, Beijing, China

2020210569@bupt.cn

Abstract. Blockchain systems, characterized by decentralization, traceability, and robust security, are increasingly adopted across various domains including finance due to their strong disaster recovery capabilities. This paper focuses on the security mechanisms underpinning Bitcoin and Ethereum, two dominant cryptocurrencies, and examines potential vulnerabilities and attack vectors targeting blockchain technologies. Specifically, it addresses attacks on consensus mechanisms, network structures, and application layers, outlining the cryptographic methods integral to blockchain security. The study delves into the types of attacks that exploit these vulnerabilities, such as forking, stale or orphaned blocks, selfish mining, and majority attacks, which threaten the integrity and consistency of blockchain systems. Additionally, it reviews the current defense mechanisms and proposes new strategies to enhance the security and resilience of these digital platforms. By exploring these security challenges comprehensively, the paper aims to provide deeper insights into the defense strategies that can fortify blockchain against such vulnerabilities, thereby enhancing its reliability and trustworthiness in digital transactions and ensuring its sustainable integration into the digital economy.

Keywords: Blockchain, Cybersecurity, network attacks.

1. Introduction

Blockchain technology has rapidly gained traction as a revolutionary distributed ledger system characterized by decentralization, traceability, and robust security features. Initially introduced in 2008 with Bitcoin, blockchain has since expanded far beyond cryptocurrency applications, finding its way into various industries, including finance, healthcare, and supply chain management, due to its inherent strengths in disaster recovery, data integrity, and trustless transactions [1]. Blockchain's decentralized nature eliminates the need for central authorities, relying instead on cryptographic techniques and consensus mechanisms to ensure data consistency across distributed nodes. As such, blockchain offers a compelling solution for environments that demand high security, transparency, and resilience against data tampering.

Despite its strengths, the growing adoption of blockchain systems has also attracted a range of attacks that exploit their inherent vulnerabilities. Major cryptocurrencies such as Bitcoin and Ethereum, which rely heavily on consensus mechanisms like Proof of Work (PoW) and Proof of Stake (PoS), are

© 2024 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

particularly susceptible to attacks targeting network structure, consensus integrity, and application layers [2]. Vulnerabilities such as forking, stale and orphaned blocks, selfish mining, and majority (51%) attacks pose significant threats to blockchain's integrity and trustworthiness. Moreover, sophisticated attacks on network layers, including DNS and BGP hijacking, disrupt node connectivity and blockchain synchronization, while smart contract vulnerabilities can lead to significant financial losses in decentralized applications [3].

This paper provides a comprehensive analysis of the security challenges facing blockchain systems, particularly focusing on Bitcoin and Ethereum as case studies. The study delves into three main areas: consensus mechanisms, network vulnerabilities, and application layer threats. It explores various attack vectors, examines how these attacks compromise blockchain security, and discusses the cryptographic methods underpinning blockchain defenses. Additionally, the paper reviews existing countermeasures and proposes new strategies to enhance the security and resilience of blockchain networks. By addressing these critical vulnerabilities, the study aims to contribute to the ongoing development of more secure and robust blockchain systems that can better withstand evolving threats in the digital landscape.

2. Relevant theories

2.1. Definition of blockchain and its development

A blockchain is essentially a distributed database recording data like transaction entries. Data is collected and divided into blocks, each of which is contains a hash pointer to a unique preceding block, such that every block dates back to the genesis block, forming a chain of blocks. Blockchain systems utilize cryptography to ensure authenticity of data so that every node can verify validity of data without a central node.

Blockchain development comes in three stages. In stage 1, Blockchain is merely the foundation of cryptocurrency. In 2008, S. Nakamoto proposed the idea of Bitcoin, which has remained the most recognized cryptocurrency to this day. The application of Bitcoin is limited since the only features it has are mining, sending and receiving coins and broadcasting signed messages. In stage 2, more complicated features are added. ETH provided users with the ability of creating smart contracts, which can be used to automate transaction, do equity-crowdfunding and security trading and do issuance. In stage 3, Essences of Blockchain such as decentralization, immutability of data and use of cryptography have gained wide adoption, and blockchain technologies are use in broader scenarios like identity verification, internet of things and voting [2].

2.2. Structure of blockchain systems

A blockchain system typically consists of five parts: The data structure, the P2P network, the Consensus mechanism, the Incentive mechanism and the Application. Smart contracts and automated scripts are optional. As show in the figure 1.

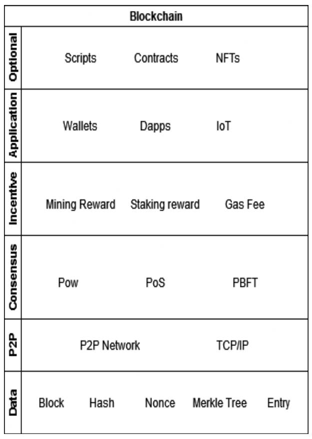

Figure 1. Blockchain layers (Photo credit: Original).

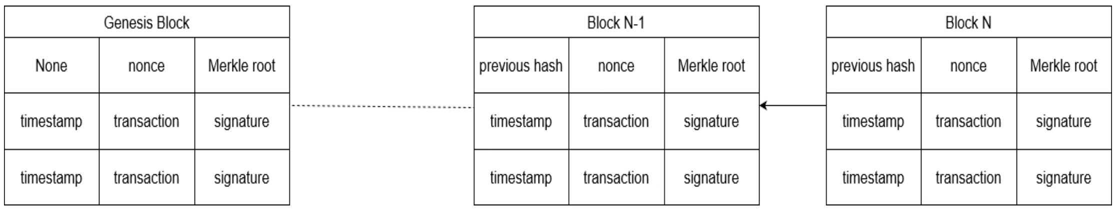

Figure 2. Data structure (Photo credit: Original).

As show in the figure 2. The data structure is how data is organized and verified in blockchain systems. Typically, blocks contain a block header and block body. The data is stored in the block body, and the header contains information such as hash of the preceding block, timestamp, a random number called nonce and root of the Merkle tree. Each transaction entry is signed by the private key of address holder.

For every block, the preceding block is unique and unchangeable. Modifying or retracting published blocks is not possible. So data on the blockchain is tamper-resistant. Merkel tree is a binary hash tree that can be used to quickly verify the integrity of large volumes of data. Blockchain data structure is cleverly designed so that every node can verify data validity independently.

The P2P network is the method of connecting nodes. It is mainly about how nodes can reach each other, how to listen and broadcast changes and how to verify data from other nodes.

The consensus mechanism is how nodes can reach agreement on current blockchain status [3]. It is mainly about how to generate new blocks and how to select a block when different valid blocks are generated at the same time. In this article I will mainly introduce the following consensus models: PoW, PoS, and PBFT.

PoW stands for Proof of work, which requires node to do computationally intensive tasks. In the famous Nakamoto consensus, nodes are required to guess a random nonce which can make the hash of the block smaller than a certain difficulty [4]. Whoever creates blocks faster is thought to have more computational resources and is trusted. When different chains are present, the network trusts the longer one since it requires more guess work.

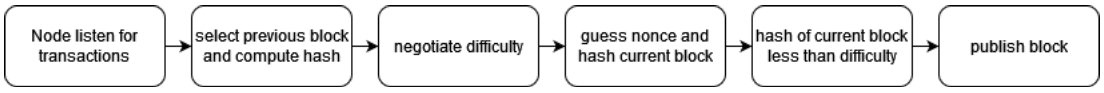
Node listen for transactions
select previous block and compute hash
negotiate difficulty
guess nonce and hash current block
hash of current block less than difficulty
publish block

Figure 3. Pow process (Photo credit: Original).

As show in the figure 3. PoS stands for Proof of stake. In PoS, nodes are required to stake their tokens. The network selects a node based on how many tokens they stake, and the winner is responsible for generating a new block. Pow removes the high demand for computation, and thus is very energy efficient [5].

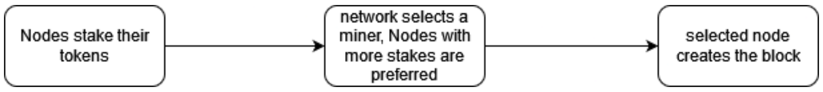
Nodes stake their tokens
network selects a miner, Nodes with more stakes are preferred
selected node creates the block

Figure 4. Pos process (Photo credit: Original).

As show in the figure 4. PBFT stands for Practical Byzantine Fault Tolerance, which is a consensus model designed to solve the Byzantine general problem. BPFT can ensure consistency among nodes so long as the number of malicious nodes accounts for less than 1/3 of the totol nodes.

The incentive layer is about rewarding nodes that produces new valid blocks. Some cryptocurrencies allows node owners to add a certain amount of coins to their wallet address while others ask people who makes transactions to pay node operators a fee. The application layer is about applications built upon or utilizing blockchain, such as wallet and Dapps.

2.3. *Security model in blockchain*

The specific security features of blockchain are: Once a block is added, it cannot be removed.

Each transaction should be cryptographically guaranteed to come from an authorized party.

Transaction history is completely public, and every transaction can be fully traced.

The blockchains held by each node, if fully synchronized, should remain consistent.

Time-stamping and cryptographic methods are used to ensure that transactions are immutable [6].

The aforementioned security is based on the security of five elements: hash functions, public-key cryptography systems, consensus mechanisms, P2P network security, and blockchain application security. Given that attacks on the first two elements are much more difficult than on the latter three, mainstream attacks on blockchain systems generally rely on targeting the latter three. Disrupting node synchronization is the primary method. Therefore, this article mainly focuses on attack on Consensus, network, application and smart contracts.

3. Vulnerabilities in blockchain systems

3.1. *Consensus layer vulnerabilities*

Vulnerabilities in consensus mechanisms refer to the potential for the network to fail to achieve consistency regarding the current state of the blockchain. One primary method of disrupting blockchain consistency is through "forking." Forking refers to the situation where multiple nodes in a blockchain network fail to reach consensus, causing the network to split into sub-networks that operate under different rules and maintain different versions of the blockchain. Forking can be caused either by malicious nodes deliberately executing conflicting validation rules or by adjustments to predetermined rules.

In addition to forking, weaker inconsistencies can occur more frequently. A Stale Block refers to a legitimate block being rejected by the network. This usually happens when multiple miners simultaneously discover different valid blocks. An Orphaned Block refers to a block whose predecessor is rejected by the network, such as when a longer chain is proven to exist; at this point, the entire network switches to the longer chain and rejects the predecessor of the currently mined block. Such inconsistencies can be exploited by attackers to amplify discrepancies between nodes, leading to larger forks.

To address these inconsistencies, the main solutions include dynamically adjusting difficulty to avoid multiple miners discovering blocks around the same time and rewarding and recording different valid blocks discovered in close temporal proximity [7]. With the increasing concentration of Bitcoin network hash power in large mining farms, both orphaned and stale blocks have been gradually decreasing in recent years.

Major Consensus layer attacks are: As show in the figure 5.

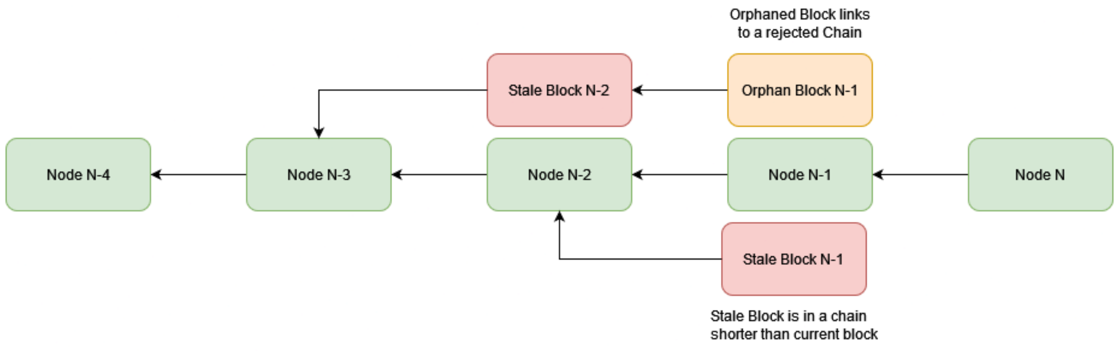
Orphaned Block links to a rejected Chain
Stale Block N-2
Orphan Block N-1
Node N-4
Node N-3
Node N-2
Node N-1
Node N
Stale Block N-1
Stale Block is in a chain shorter than current block

Figure 5. Stale & Orphaned Blocks (Photo credit: Original).

3.1.1. Selfish mining. Selfish mining refers to a scenario where a node, after computing the next block (bn+1) following the current block (bn), does not immediately publish it. Instead, it continues to compute the subsequent blocks (bn+2, bn+3). During this process, other nodes might compute bn+1, bn+2, and so on. However, since the selfish node's chain is sufficiently long, once it publishes its chain, the network will accept it, causing the blocks computed by other nodes (bn+1, bn+2, etc.) to become Stale or Orphaned Blocks. This increases network inconsistency and can even lead to a Fork [8].

To counteract selfish mining, the following strategies can be employed:

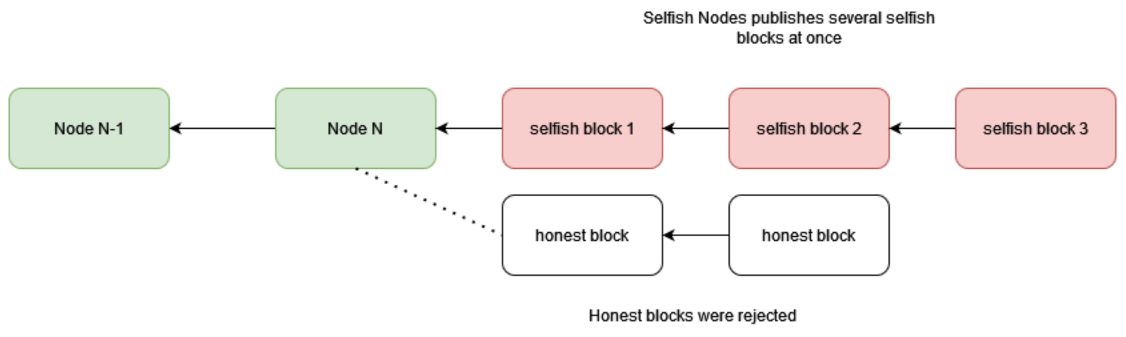

Figure 6. Selfish mining (Photo credit: Original).

As show in the figure 6. Set deadlines for each block. If a block is not broadcast to the network within the specified time, it is discarded.

Introduce mechanisms that reward the publication of newer blocks more than multiple older blocks published at once, reducing the profit potential for attackers.

Develop methods to distinguish between blocks from selfish nodes and those from honest nodes. A way is to detect the difference between the block height of the subsequent block and the average expected confirmation height of all transactions in the previous block. Saad et al. also have designed a feasible solution [9].

3.1.2. Majority attack (51% Attack). A majority attack occurs when an attacker controls more than 50% of the network's hashing power. With this majority, the attacker can use their computational advantage to ensure that their chain is always adopted by the network, rendering blocks mined by others invalid. The most notorious application of a majority attack is double spending. In this scenario, the attacker first performs a transaction, then uses their hashing power to recalculate a chain that does not include the recent transaction. Due to the hashing power advantage, the latter chain will eventually be accepted by the network as valid, effectively reversing the transaction, and undermining the blockchain system's ideal of immutability.

In practice, with the advent of NiceHash and large mining farms, it is not uncommon for a single group to gain a temporary hashing power advantage. Additionally, a successful majority attack does not necessarily require the attacker to possess more than 50% of the hashing power. Suppose an attacker holds a proportion x of the total hashing power (0 < x < 1), and double spending requires k blocks for confirmation. The probability of the attacker's success is:$1 if x > 0.5, \left(\frac{x}{1-x}\right)^k if x \leq 0.5$.

There are some defense mechanism against 51% attack. One method is that honest miners can calculate the historically weighted difficulty (HWD) and choose a chain with higher HWD. This works because when an attacker concentrates his/hers computing power the block generation speed is drastically increased, and the HWD of the attacker drops. The network can also punish attacker by temporarily raising the difficulty.

3.2. Network layer vulnerabilities

The primary objective of attacks on P2P networks is to disrupt the connectivity between honest nodes. The general strategy involves deceiving or preventing other nodes from connecting to the network, thereby compromising P2P network synchronization.

3.2.1. DNS Attacks. In a P2P network, node-to-node connections rely on the TCP/IP protocol. Attackers can exploit vulnerabilities in the TCP/IP protocol to deceive nodes. DNS attacks exploit the fact that new nodes need to perform a bootstrap process to obtain the latest blockchain copy and information about other nodes. The DNS protocol is commonly used to resolve addresses to bootstrap servers. Attackers can use DNS poisoning and other methods to prevent new nodes from connecting to the network, causing them to obtain incorrect blockchain copies or connect only with malicious nodes.

3.2.2. BGP attacks. BGP attacks represent a more advanced form of attack. When the network operator is also the attacker, they can force traffic from nodes within their controlled network to be routed through malicious nodes. This can achieve goals such as concentrating hashing power, deceiving nodes, or segmenting the network.

3.2.3. Solutions. To mitigate DNS attacks, if a trustworthy DNS provider exists, clients can use encrypted methods such as DNS over HTTPS (DoH) or DNS over TLS (DoT) to ensure reliable resolution results. To counter BGP attacks, the best approach is to increase network diversity by distributing nodes across different countries and different network operators.

3.3. Application layer threats

Double spending refers to the act of a payer transferring the same amount of cryptocurrency to multiple different recipient accounts. Due to the decentralized nature of blockchains, different recipients might observe different blockchain states. In the presence of race conditions, such as Stale/Orphaned Blocks described in section 3.1, or during a majority attack, multiple recipients might see new blocks with differing states and believe they have received the transfer. When combined with attacks such as BGP attacks or DNS attacks, attackers can significantly increase the success rate of double spending attacks. In 2013, attackers used a soft fork in the Bitcoin network to perform a double spending attack.

A double spending attack based on DNS/BGP attacks involves broadcasting the same transfer to multiple targeted nodes that, due to DNS/BGP poisoning, have failed to synchronize with the network. These nodes would believe they have received a legitimate transfer while not realizing that the same funds have already been spent on another chain recorded by others.

For clients, the most effective way to counter double spending attacks based on 51% attacks is to increase the number of blocks required to confirm a transaction. As described in section 3.1.2, the larger the number of blocks k required for confirmation, the lower the probability of an attacker's success. Clients should also adopt the solutions outlined in section 3.2.3 to address DNS/BGP attacks, ensuring synchronization with the network.

3.4. Smart contract vulnerabilities

Smart contracts are one of the major innovations of Blockchain 2.0. They allow users to write custom code to execute their applications on the blockchain. Smart contracts can create agreements between multiple untrusted parties without needing a centralized third party to enforce the contract [10]. The most well-known smart contracts are Ethereum smart contracts, where users can write contracts in the Solidity programming language to run on the Ethereum blockchain. However, like other programming languages, smart contract code can have exploitable vulnerabilities if not carefully reviewed.

Re-entry Vulnerability: This occurs due to a race condition where an external attacker can call the same or different functions before the current contract call returns, allowing the attacker to insert malicious code.

Permission Control Vulnerability: Some contract code does not adequately check permissions, allowing attackers to gain control over the contract.

Front-Running Attack: Ethereum uses Gas values to set transaction fees paid to miners. Higher Gas values will prompt miners to prioritize the transaction. Some attackers monitor unconfirmed smart contracts and then set higher Gas values to execute the same contract content, completing the transaction earlier.

Arithmetic Overflow Vulnerability: Arithmetic calculations in smart contracts can overflow. For instance, transactions using fixed-length numeric types for conditional checks may result in contract malfunctions due to arithmetic overflow.

Exception Handling Vulnerability: Contracts that do not handle exceptions or error return values properly can be vulnerable to exploits.

Denial of Service Vulnerability: If an exception occurs within a loop, the entire loop operation may roll back, causing significant disruptions or even making the contract unusable. Attackers can exploit this to prevent the smart contract from functioning.

Short Address Attack: Ethereum smart contracts use 20-byte addresses for parameter passing. If the parameter length is incorrect, the Ethereum Virtual Machine automatically pads it with zeros. Attackers can craft specific short addresses (less than 20 bytes), leveraging the VM's padding mechanism to send incorrect amounts of Ethereum to the wrong addresses.

Most vulnerabilities in smart contracts are related to the contract syntax, the way the contract is written, and the execution environment of the contract's virtual machine. Liu Zhexu et al. researched methods for automatically and intelligently detecting vulnerabilities in contract code. Another approach is to use languages with strong static checking, mandatory error handling, and constraints on race conditions, such as Rust, to write smart contracts.

3.5. Privacy issues

Traditional blockchains offer good traceability, but every transaction's history is fully recorded and publicly accessible. Ideally, addresses on the blockchain should not be linked to real-world identities. However, with the rise of exchanges and cryptocurrency payment, addresses on the blockchain are increasingly linked to real-world identities. In such cases, the fully public nature of the blockchain can lead to privacy issues.

Some new cryptocurrencies, such as Zcash and Monero, attempt to address privacy issues using new encryption algorithms and obfuscation techniques. Zcash combines multiple transactions into one, obscuring the transactions to provide better privacy, while Monero introduces Ring-Signature, an algorithm that can verify signatures without knowing the public key of the signer. Another solution is to allow the blockchain to not record all transactions, enabling some transactions to occur off the blockchain. This is known as Off-chain Function.

4. Case studies

In this chapter the author will go over major real work attacks and their methods.

The Mt.Gox crypto exchange was attacked several times and led to its bankruptcy. The attackers gathered useful information from Bitcoin’s Blockchain and engineered a fake transaction ripple to increase the market price. They utilized a software in Bitcoin software to inject fake transaction IDs.

In 2014, May and June 2018, five Blockchain-based cryptocurrencies; namely, Monacoin, Bitcoin Gold, Zencash, Verge, and Litecoin Cash, were targeted by a 51% attack.

In August in 2014, a malicious ISP in Canada announced BGP prefixes belonging to major ISPs including Amazon, OVH, Digital Ocean, LeaseWeb, and Alibaba, and intercepted the traffic routed to mining pools.

In April 2018, BGP attacks were launched against MyEtherWallet.com, an open source web application used for exchanging Ethereum tokens online. Attackers managed to steal 152,000 USD from the web application.

A hacker exploited one smart contract of Ethereum called DAO’s code weakness and stole more than 50 million USD worth of crypto-currency reported on June 17, 2016.

In January 2018, a hacker discovered a bug of integer overflow with smart contracts using in Proof of Weak Hands (PoWH) coin and stole 888 ETH. In October 2018, an attacker launched a reentrancy attack targeted at smart contracts of Spankchain and drained 165.38 ETH.

5. Conclusion

This paper provides a comprehensive analysis of the security challenges within blockchain systems, focusing on consensus mechanisms, network vulnerabilities, and application layer threats. By examining the cases of Bitcoin and Ethereum, the study identifies critical attack vectors such as forking, stale and orphaned blocks, selfish mining, majority attacks, and various smart contract vulnerabilities. The cryptographic methods and defense mechanisms underpinning blockchain security are also discussed, highlighting the need for robust strategies to mitigate these risks. The findings underscore that the primary vulnerabilities in blockchain systems arise from inconsistencies in consensus protocols, network disruptions, and exploitable flaws in smart contracts. Addressing these issues is crucial for enhancing the security, reliability, and broader adoption of blockchain technologies in digital transactions and other applications. Looking ahead, future research should focus on developing more resilient consensus mechanisms, such as hybrid PoW/PoS models, to balance security, scalability, and energy efficiency. Enhancing smart contract security through advanced static analysis, automated vulnerability detection, and the adoption of safer programming languages like Rust could significantly reduce risks. Additionally, the integration of privacy-enhancing technologies, including Ring-Signature and Zero-Knowledge Proofs, is essential to address privacy concerns in transparent blockchains. Further exploration into decentralized applications, particularly those relying on blockchain 2.0 systems like DApps and smart contracts, should aim to create legally sound and secure frameworks. By advancing these areas, the blockchain community can build more robust systems capable of withstanding evolving cybersecurity threats while maintaining the trust and integrity that are foundational to blockchain's success.

References

[1] König L, Unger S, Kieseberg P, Tjoa S and JRC Blockchains 2020 The risks of the blockchain: a review on current vulnerabilities and attacks J. Internet Serv. Inf. Secur. 10(3) 110–127

[2] Duan L, Sun Y, Ni W, Ding W, Liu J and Wang W 2023 Attacks against cross-chain systems and defense approaches: a contemporary survey IEEE/CAA J. Automatica Sinica 10(8) 1647–1667

[3] Dwivedi K, Agrawal A, Bhatia A and Tiwari K 2024 A novel classification of attacks on blockchain layers: vulnerabilities, attacks, mitigations, and research directions arXiv preprint arXiv:2404.18090

[4] Homoliak I, Venugopalan S, Reijsbergen D, Hum Q, Schumi R and Szalachowski P 2020 The security reference architecture for blockchains: toward a standardized model for studying vulnerabilities, threats, and defenses IEEE Commun. Surv. Tutor. 23(1) 341–390

[5] Zhu X, Huang Y, Wang X and Wang R 2023 Emotion recognition based on brain-like multimodal hierarchical perception Multimedia Tools and Applications 1–19

[6] He D, Deng Z, Zhang Y, Chan S, Cheng Y and Guizani N 2020 Smart contract vulnerability analysis and security audit IEEE Netw. 34(5) 276–282

[7] Chaganti R, Boppana R V, Ravi V, Munir K, Almutairi M, Rustam F, Lee E and Ashraf I 2022 A comprehensive review of denial of service attacks in blockchain ecosystem and open challenges IEEE Access 10 96538–96555

[8] Guggenberger T, Schlatt V, Schmid J and Urbach N 2021 A structured overview of attacks on blockchain systems PACIS 100

[9] Iqbal M and Matulevičius R 2021 Exploring sybil and double-spending risks in blockchain systems IEEE Access 9 76153–76177

[10] Ivanov N, Li C, Yan Q, Sun Z, Cao Z and Luo X 2023 Security threat mitigation for smart contracts: a comprehensive survey ACM Comput. Surv. 55(14s) 1–37