Smart Contract Vulnerabilities: A Literature Review

Runesh Bhardwaj[1], Naveen Kumar[2], Ritesh Rana[3], Sandeep Kumar[4], Ashok Kumar Kashyap[5]
[1, 3]Dept of Computer Science
[2]Assistant Professor, Dept of Computer Science
[4,5]Assistant Professor (CS), Dept of Computer Science
[1, 2, 3] Himachal Pradesh University, Shimla
[4, 5]ICDEOL, Himachal Pradesh University, Shimla

Abstract- *An innovative approach to data storage, transactional executions, multitasking and trust in an open environment is offered by blockchain. It is a breakthrough technology for cryptography and cybersecurity has been used in various use cases such as globally deployed cryptocurrency systems like Bitcoin and smart contracts. However, despite the growing interest in blockchain technology in recent years, the security and privacy of blockchains continue to be the centre of debate while implementing blockchain technology in various applications. In this paper, a comprehensive overview over the security of blockchain is presented. The basic security properties, essential requirements and building blocks for blockchain are described, followed by other vulnerabilities in smart contracts. Finally, the vulnerabilities in smart contracts are reviewed, including Reentrancy vulnerability, Arithmetic overflow and Underflow vulnerability, Timestamp Dependency vulnerability, Access Control vulnerability and so forth. This literature review can be used as a vital starting point for the readers to gain an in-depth understanding of the attributes, techniques, privacy and security concepts of smart contracts in blockchain systems.*

Keywords- Blockchain, Decentralisation, Smart contracts, Vulnerabilities, Review, Survey

I. INTRODUCTION

Over time, the concept and technology of computing and storing data have changed drastically. From carrying 4KB RAM to the first ever moon landing mission in 1969 on Apollo 11 to capturing a black hole image of size 4 million gigabits using very-long-baseline interferometry (VLBI) in 2019, the ways and concepts of storing data have changed drastically. Initially, magnetic tape and punched cards were used to store data on computers. These technologies were slow, bulky and unreliable. Over time, new storage technologies such as Magnetic core memory and transistorized memory were developed that offered greater speed, space and reliability.Solid-state memory, hard disk drives and flash memory followed. With new technologies emerging, the use of old technologies was reduced, ultimately terming some of the old technologies as obsolete. Currently, online storage technology or cloud computing has revolutionised the concept of storing and accessing data by providing users with extended online storage and working space. The newest addition to online storage technology is blockchain technology. Blockchain technology has not only revolutionised the concept of storage but has also impacted the state of finance worldwide.

BLOCKCHAIN: In 2008, the whitepaper Bitcoin: A Peer-to-Peer Electronic Cash System was created and released by Satoshi Nakamoto an anonymous identity to date. The whitepaper described a "purely peer-to-peer version of electronic cash" called Bitcoin and marked the public debut of blockchain technology.In simple terms, blockchain is a distributed ledger (transaction record) technology that records and shares information in a decentralized manner making it transparent, immutable and secure.The basic characteristics such as decentralisation, transparency, immutability and security of Blockchain technology have made it one of the most effective solutions for a range of applications. Whereas on the technical bit blockchain is a distributed ledger technology that works on the principle of consensus algorithms written on smart contracts and allows the users to verify the transactions by mining[1].

EVOLUTION OF BLOCKCHAIN: Chaum is known to introduce blockchain-like protocol in his Ph.D. thesis for the very first time in 1982[2]. Further Haber and Stornetta introduced a secured chain of blocks cryptographically in 1991 [3]. Bayer et al. later improvised the technology and added Merkle Tree to the design in 1993 [4]. Nick Szabo designed" Bit Gold", the first decentralised digital currency mechanism in 1998 which was the first practical testimony as a technology[5]. Finally, a breakthrough for the public came in 2008 with the introduction of Bitcoin, a complete peer-to-peer network-based electronic cash system[1]. The term "Blockchain" was also introduced in 2008 as a module behind the Bitcoin transaction [6]. Buterin proposed Ethereum in his whitepaper in 2013. The growth of Ethereum was crowdfunded in 2014 and the Ethereum network was made public on July 30, 2015. The rise of Ethereum symbolised the development of blockchain 2.0[7].

Working and Structure of Blockchain: Blockchain is a method of storing data in a block that is linked to the previous and the next block. The structure is more like a chain of blocks that are linked together to form secure immutable data storage [8]. It can be understood on the bases of the following steps:

1. **Structure of a block:** The block structure mainly consists of 3 items:
   a. **Data:** The transactions in a blockchain system are referred to as data.
   b. **A unique fingerprint as Hash:** Hash is referred to as a unique combination of letters and numbers. It is like a fingerprint for the data stored in a block and is always exclusive to each block in the Blockchain. The hash will also be altered when the data changes within a block.
   c. **Previous Blocks Hash:** The previous block's hash is contained within a block, thus creating a chained structure. When a transaction in any block is altered, the hash of that block will also change.
2. **P2P Network:** A peer-to-peer (P2P) network is a decentralized network where all nodes have equal power and communicate with each other directly.Peer-to-Peer network is mainly used in blockchain verification where each node in the network checks and verifies if all transactions in a block are valid before adding it to the blockchain, ensuring the security and integrity of the data. This process is known as consensus.
3. **Transactions:** Blockchain is provided with a public key and a private key by computer software. The user must withhold his private key as the private key is used to signee the digital transactions. The public key can be revealed to everyone and all the digital signatures can be verified using the corresponding public key.

Types of Blockchain: Blockchain technology is a distributed ledger technology that provides a secure and transparent way of recording data.There are four main types of blockchain: public, private, consortium and hybrid.

1. **Public blockchain:** It is a decentralized network where anyone can join and participate, making it accessible to the public. **Bitcoin** is the most familiar example of public blockchain technology.
2. **Private blockchain**: Private key is a permissioned network where access is restricted to a particular group of users. These blockchains are typically used for internal purposes, such as supply chain management in an organization.
3. **Consortium blockchain:** It is a permissioned network where multiple organizations control the nodes. These blockchains are often used in industries where multiple parties need to collaborate on a shared system, such as finance or healthcare.
4. **Hybrid blockchain:** It is a combination of public and private blockchains. In a hybrid blockchain, some data is stored on a public blockchain, while the remaining data is stored on a private blockchain.

II. SMART CONTRACTS

Smart contracts are self-executing agreements that run on a blockchain network. They are programmed using computer codes and contain a set of rules, which determines, how a contract will be executed. Once the contract conditions fulfil, the smart contract automatically executes without any intervention from the third party.

Characteristics of Smart Contract: Smart Contracts provide a secure and automated way to execute transactions on a blockchain network. They are written in specialized programming languages such as Solidity or Vyper and their code is immutable once deployed on the network. Smart contracts are executed in a decentralized manner, with each node on the network validating and executing the contract. Gas fees are required depending on the consensus mechanism to cover the cost of executing the smart contracts and they can interact with other smart contracts to create complex decentralized applications.

1. **Programming Languages for Smart Contracts:** Certain Programming languages are specifically designed to write smart contracts on the blockchain. Solidity is the most used language for creating smart contracts on the Ethereum blockchain, while Vyper is another popular language that aims to simplify the process of creating secure smart contracts.
2. **Immutability of Smart Contract Code:** Smart contract code is immutable, meaning, it cannot be altered once it has been deployed on the blockchain. This ensures that the contract will be executed as intended, without the risk of interference or tampering.
3. **Decentralized Execution of Smart Contracts:** Smart contracts are executed in a decentralized manner on a network of nodes rather than a central server. This provides security and transparency, as each node in the network validates and executes the contract.
4. **Self-Execution of Smart Contracts:** Smart contracts are self-executing, meaning, they automatically execute their programmed instructions when predefined conditions are met. This automation eliminates the need for intermediaries, reduces the risk of errors and speeds up the execution process.

5. **Transaction-Based Execution of Smart Contracts:** Smart contract execution is triggered by transactions that are submitted to the blockchain network. Transactions can be triggered by different types of events, such as the receipt of funds, the occurrence of a specific date or time, or the fulfilment of specific conditions.
6. **Contract Storage on the Blockchain Network:** Smart contracts have their own storage space on the blockchain network, where they store data related to the contract's execution. This storage space is accessible only to the smart contract itself and is maintained by the blockchain network.
7. **Payment of Gas Fee for Smart Contract Execution:** To execute the smart contract on a blockchain network transaction fee must be paid, this fee is known as a gas fee. The gas fee is paid in cryptocurrency of the underlying blockchain network. The gas fee covers the cost of processing the transaction and executing the smart contract.
8. **Interoperability of Smart Contracts:** Smart contracts can be designed to interact with one another, creating a network of contracts that can exchange data and execute complex workflows. This interoperability enables the creation of decentralized applications (DApps) that run entirely on the blockchain network.

Architecture of Smart Contracts: A smart contract is a computer program that runs on a blockchain network, allowing parties to execute an agreement without the need for intermediaries. The structure of a smart contract consists of three main parts: the agreement code, the data and the functions.

1. **The Agreement Code:** The agreement code defines the terms of the contract and the conditions for execution. The agreement code outlines the rules and requirements that the parties must follow and determines the circumstances under which the contract will be executed.
2. **The Data:** The data component of a smart contract includes the input parameters and the output result of the contract. The input parameters include the data provided by the parties to initiate contract execution, such as payment amounts and deadlines. The output results are the results of the contract's execution.
3. **The Functions:** The actions that a smart contract can perform are defined by the functional component of a smart contract. These actions can include transferring funds, updating data on the blockchain and executing other tasks necessary for the contract's execution.

Working of Smart Contracts: Smart contracts have a similar but not exact workflow as the normal scripts. **The working of smart contracts can be explained based on the following steps:**

1. **Identify the need for a smart contract:** The first step is to identify the need for a smart contract. A smart contract is a self-executing contract, coded on a blockchain. Smart contracts are used when there is a need for automation of a contract.
2. **Define the contract terms and conditions:** The second step is to define the terms and conditions of the contract. This includes the conditions that must be met for the contract to be executed.
3. **Choose a blockchain platform:** The third step is to choose a blockchain platform to host the smart contract. There are several blockchain platforms available, such as Ethereum, EOS and TRON.
4. **Write the smart contract code:** The fourth step is to write the smart contract code. This is done using a programming language that is supported by the blockchain platform chosen. The code must be written to execute the terms and conditions of the contract automatically.
5. **Test the smart contract:** The fifth step is to test the smart contract. This is done to ensure that the code is error-free and executes as intended. Testing involves simulating the execution of the contract and checking that the results are as expected.
6. **Deploy the smart contract:** The sixth step is to deploy the smart contract on the chosen blockchain platform. This involves sending the code to the platform, which then creates a new instance of the contract on the blockchain.
7. **Executing the smart contract:** The seventh step is to execute the smart contract. This is done by sending a transaction to the blockchain that triggers the execution of the contract.
8. **Monitor the smart contract:** The final step is to monitor the smart contract to ensure that it is executing as intended. This involves checking the blockchain to see if the contract has been executed and verifying that the results are as expected.

Application of Smart Contracts: Smart contracts have numerous applications across various industries. **The following are commonly known applications of Smart Contracts:**

1. **Supply Chain Management:** Smart contracts can be used in automation and streamlining the supply chain process.
2. **Real Estate:** Smart contracts can automate the process of buying and selling real estate. They can reduce the need for intermediaries such as brokers and lawyers.

3. **Insurance:** Smart contracts can be used for automating insurance claims. They can verify claims and automatically pay out claims when the conditions of the contract are met.
4. **Finance:** Smart contracts can be used for automating financial transactions. They can be useful in executing trades, transferring funds and enforcing financial agreements.

III. VULNERABILITIES IN SMART CONTRACT

The vulnerabilities in smart contracts can be categorised into three main levels. Where they can be found as Solidity level, EVM bytecode and Blockchain[9].

1. Solidity Level

Most of the major vulnerabilities occur at the solidity level, based on the attack incidents and economic loss, these vulnerabilities can be further classified into 2 broad categories (Heavy harm and light harm) [10].

Heavy Harm Vulnerabilities

- **Integer Over/Underflow:** In smart contracts, integer overflow and underflow vulnerabilities can occur when the maximum or minimum limit of an integer variable is exceeded, leading to unexpected behaviour and potential security threats.
- **Re-entrancy Problem:** The Re-entrancy problem, a potential vulnerability in smart contracts, can allow malicious contracts to execute unexpected behaviour and gain access to unauthorized funds or data.
- **Greedy Contract:** The greedy contract vulnerability is a security issue that can occur in smart contracts when malicious actors consume excessive resources, potentially causing a denial of service and unnecessary fees. The developers should focus on setting appropriate fees and rewards, limiting gas consumption and implementing checks to prevent abuse.

Light Harm Vulnerabilities

- **Denial of Service:** DoS attacks can occur in Ethereum smart contracts due to three types of vulnerabilities, including the dependence on external functions, gas limit constraints and programming mistakes made by developers.
- **Gas Overspent:** Sometimes a user may send more gas than required, which can lead to the user paying more than intended for the transaction and this can be exploited by attackers to drain funds from the contract.
- **Transaction Ordering (TOV):** TOV in a smart contract can be exploited by a malicious user who manipulates the order in which the transactions are processed by the network.
- **Prodigal Contracts:** It is a type of vulnerability in Ethereum smart contracts. The vulnerability is related to the automatic refund feature of the smart contract, which is designed to return the funds to the contract owner in case of an attack. However, in some cases, the smart contract may transfer funds to someone who is not related to the contract, which can result in economic losses.
- **Exception Handling:** Exception handling vulnerability can be caused by improper handling of errors or exceptions in smart contract code, which can lead to unexpected behaviour and security issues.
- **Destroyable/Suicidal Contract:** A contract in which the consumer signs a pack agreeing to give up all his crypto belongings if the contract gets destroyed, but contracts can also be destroyed by others to cause the transaction to fail.

2. Ethereum Virtual Machin (EVM) Level

The EVM uses a stack-based architecture, which means that it stores data in a stack-like structure. The maximum depth of this stack is 1024, which means that it can only hold a certain number of items at a time. Vulnerabilities in EVM can be classified as follows:

- **Call Stack Depth Limitation: T**his vulnerability allows attackers to overload the call stack of a smart contract by repeatedly calling themselves before calling another smart contract. This can cause the program to crash and potentially lead to economic losses.
- **Unchecked and Failed to Send:** This vulnerability occurs when a smart contract uses the "send" instruction to transfer ether (cryptocurrency) to another user or smart contract, but the transfer fails due to either insufficient balance or exceeding the gas limit.
- **Short Address:** In EVM, transaction information is represented as a string of bytecodes, which includes a 32-byte target Ethereum address and a 32-byte token amount. The attacker deliberately omits the last bit of the target address.
- **Authentication Through tx.origin:** The vulnerability in smart contracts related to the use of the global variable "tx.origin" can be exploited by attackers to bypass authentication checks and impersonate legitimate users.

Blockchain Level

The smart contracts are stored in a distributed consensus environment and inherit the characteristics of blockchain, which are open, transparent, immutable **and** permanently operational.

a) Block Time Dependency: Some smart contracts are dependent on the timestamp of the current block for proper execution, which is determined by the packing node with some flexibility. However, attackers can exploit this flexibility by setting their timestamps to manipulate smart contract results related to timestamps.

b) Unsecured Balance: This vulnerability occurs when the ether balance of a smart contract is made accessible to attackers. Ether is the cryptocurrency used on the Ethereum blockchain **and** smart contracts can hold and manipulate ether as well as other valuable assets.

IV. LITERATURE REVIEW

1. **S. Nakamoto** proposed a solution to the double-spending problem in electronic payment systems using a peer-to-peer distributed timestamp server. The system is based upon cryptographic proof instead of trust, allowing any two willing parties to transact directly without needing a trusted third party. The system is secure if honest nodes collectively control more CPU power than any cooperating group of attacker nodes. The proposed system uses digital signatures and hash-based proof-of-work to prevent fraud, protect sellers, require minimal structure and allow nodes to leave and re-join the network at will. Once the latest transaction in a coin is buried under enough blocks, the spent transactions before it can be discarded to save disk space. Suggestions arise that routine escrow mechanisms could easily be implemented to protect buyers [1].
2. **B. Jiang et al.** presented ContractFuzzer, a tool for testing Ethereum smart contracts for security vulnerabilities. Experimental results of a study were discussed in which ContractFuzzer was used to test 6,991 smart contracts and found 459 vulnerabilities with high precision. The conclusion states that ContractFuzzer is an effective tool for detecting security vulnerabilities in Ethereum smart contracts and highlights the need for more research on smart contract security. Outcomes may act as a strong starting point for several future works, including, improving the efficiency of ContractFuzzer, extending it to support other blockchain platforms and developing a tool to automatically repair vulnerabilities detected by ContractFuzzer [11].
3. **N. Kshetri** evaluated the role of blockchain in strengthening cybersecurity and protecting privacy, with a focus on its impact on Internet of Things (IoT) security. It compares blockchain with cloud storage in terms of security and privacy. Also, discussing the use of blockchain in managing supply chains and its potential to deliver real ROI. Additionally, highlighting the potential of blockchain-based identity and access management systems in addressing IoT security challenges. The authors provide practical applications and real-world examples to support their arguments. The paper also discusses policy implications related to the use of blockchain in IoT security and privacy [12].
4. **X. Zhang and X. Chan** proposed a data security sharing and storage system based on the consortium blockchain for vehicular ad-hoc networks. The proposed system uses digital signature techniques based on bilinear pairing for elliptic curves to ensure data reliability and integrity. The consortium blockchain technology provides a decentralized, secure and reliable database maintained by the entire network node. Smart contracts are used to limit triggering conditions for preselected nodes when transmitting and storing data and for allocating data coins to vehicles that contribute data. Also suggests that blockchain technology can provide a secure and reliable solution for data sharing and storage in VANETs. The use of digital signature techniques and smart contracts can further enhance the security and efficiency of the proposed system. However, there are still challenges that need to be addressed, such as the scalability and cost of deploying blockchain-based solutions in large-scale VANETs [13].

V. COMPARATIVE ANALYSIS

Key Findings and the limitations of the previous studies

Year	Study Title	Research Objective	Method Used	Sample Characteristics	Key Findings	Limitations	Implications
2019	**Smart Contract Vulnerabilities: Does Anyone Care?**[14]	Survey the vulnerabilities in smart contracts	Survey	21,270 vulnerable contracts	At most 504 out of 21,270 contracts have been subjected to exploits. Vulnerable code impact had been exaggerated.	does not consider unreported vulnerabilities	Highlight the need to shift the focus from identifying vulnerabilities to developing solutions to eliminate them.
2020	**An Analysis of Ethereum Smart Contract Vulnerabilities** [15]	Identify and examine the ten most common vulnerabilities in smart contracts	Data analysis	Smart contracts deployed on the Ethereum blockchain in 2020	The vulnerabilities emerging critical attacks that exploit the vulnerabilities and what can be done to avoid them.	Only focuses on the vulnerabilities in Ethereum smart contracts in 2020.	Provide a threat landscape for 2020 and suggest measures to avoid vulnerabilities.
2021	**Blockchain Vulnerabilities in Practice**[16]	Describe vulnerabilities observed in smart contracts and node software and how to avoid them	Field note	Ethereum ecosystem	Detailed information on Coreblockchain and smart contract vulnerabilities and how to avoid them.	Only focuses on smart contracts and core blockchain vulnerabilities.	Provides information on observed vulnerabilities in smart contracts and node software and suggests measures to avoid them.
2021	**A Survey on Security Verification of Blockchain Smart Contracts**[17]	Review the security verification of blockchain smart contracts	Literature review	53 most related papers	20 papers focus on the security assurance of blockchain smart contracts and 33 papers focus on the correctness verification of blockchain smart contracts.	The study only focuses on the security verification of blockchain smart contracts.	Provides a taxonomy of security verification of blockchain smart contracts and discusses the pros and cons of each category of related studies.
2021	**Trends in publishing blockchain surveys: a bibliometric perspective**[18]	Analyse the trends in publishing blockchain surveys	Bibliometric analysis	801 surveys or review papers published in the field of blockchain	Provides interesting insights into the publication type, publishers and venue, references, citations, paper length, various	The study only focuses on the trends in publishing blockchain surveys.	Provides insights into the trends in publishing blockchain surveys.

					categories, year, countries, authors and their collaborations.		
2021	**A Literature Survey on Smart Contract Testing and Analysis for Smart Contract Based Blockchain Application Development**[19]	Survey the techniques and approaches discussed in various selected related papers	Literature review	Selected research studies	Identified open challenges that require further research.	The study only focuses on the techniques and approaches discussed in various selected related papers.	Provides a comprehensive survey on smart contract testing and analysis of smart contract code.
2022	**A Deep Dive into Blockchain-based Smart Contract-specific Security Vulnerabilities**[20]	Discuss security issues in smart contract applications	Literature review	Smart contracts deployed on different blockchain platforms	Categorized the vulnerabilities into smart contract platform, applications that integrate with the blockchain and the vulnerabilities in smart contract code.	The study only focuses on security issues in smart contract applications.	Provides detailed information on smart contract-specific vulnerabilities and their defence.
2022	**Detecting Vulnerabilities in Smart Contract within Blockchain: A Review and Comparative Analysis of Key Approaches**[21]	Critically review and analyse key approaches for detecting vulnerabilities in smart contracts within Blockchain	Literature review	Five key approaches	Comparison of five key approaches for detecting vulnerabilities in smart contract within Blockchain.	The study only focuses on the detection of vulnerabilities in smart contract within Blockchain.	Provides a critical review of key approaches for detecting vulnerabilities in smart contract within Blockchain.
2022	**Systematic Review of Security Vulnerabilities in Ethereum Blockchain Smart Contract**[22]	Systematically review security vulnerabilities in Ethereum blockchain-based smart contract	Systematic review	Security vulnerabilities in Ethereum blockchain-based smart contracts	Discussed Ethereum smart contract security vulnerabilities, detection tools, real-life attacks and preventive mechanisms.	The study only focuses on security vulnerabilities in Ethereum blockchain-based smart contracts.	Provides a comprehensive review of security vulnerabilities in Ethereum blockchain-based smart contract.

2023	**Smart Contract Security: A Software Lifecycle Perspective**[23]	Review smart contract security from a software lifecycle perspective	Literatur e review	Smart contracts deployed on different blockchain platforms	Summarized the common security vulnerabilities of smart contracts and examined recent advances in smart contract security spanning four development phases.	The study only focuses on smart contract security from a software lifecycle perspective.	Provides a comprehensive review of smart contract security from a software lifecycle perspective.

VI. CONCLUSION

A comprehensive overview of vulnerabilities in blockchain-based smart contractsare reviewed here. It highlights that although thousands of vulnerabilities have been identified, only a small fraction have been exploited in real-world attacks. Common vulnerabilities such as reentrancy and integer overflow were categorized, leading to significant financial losses. The review delves into security verification methods, vulnerability detection, and comparative analysis of detection techniques. It also suggests best practices including adherence to secure coding standards, input validation, gas limit settings, and the use of security tools. However, it notes limitations like a focus on reported vulnerabilities, specific platform (e.g., Ethereum) coverage, and limited analysis years, emphasizing the need for ongoing research and solution development. Robust security is crucial for the widespread adoption of blockchain and smart contracts in domains like finance, healthcare, and supply chain, and this review serves as a foundation for future security-enhancing research in blockchain systems.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[2] D. L. Chaum, *Computer Systems established, maintained and trusted by mutually suspicious groups*. Electronics Research Laboratory, University of California, 1979.

[3] S. Haber and W. S. Stornetta, "How to time-stamp a digital document," *Journal of Cryptology*, vol. 3, no. 2, pp. 99–111, 1991, doi: 10.1007/BF00196791.

[4] D. Bayer, S. Haber and W. S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," in *Sequences Ii*, Springer, 1993, pp. 329–334.

[5] "Bit Gold Satoshi Nakamoto Institute."

[6] "A timeline and history of blockchain technology."

[7] "Ethereum Whitepaper ethereum.org."

[8] "Blockchain — A Short and Simple Explanation with Pictures HackerNoon."

[9] N. Atzei, M. Bartoletti and T. Cimoli, "A Survey of Attacks on Ethereum Smart Contracts (SoK)," in *Principles of Security and Trust*, M. Maffei and M. Ryan, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 164–186.

[10] X. Tang, K. Zhou, J. Cheng, H. Li and Y. Yuan, "The vulnerabilities in smart contracts: A survey," in *Advances in Artificial Intelligence and Security: 7th International Conference, ICAIS 2021, Dublin, Ireland, July 19-23, 2021, Proceedings, Part III 7*, Springer, 2021, pp. 177–190.

[11] B. Jiang, Y. Liu and W. K. Chan, "ContractFuzzer: Fuzzing smart contracts for vulnerability detection," in *ASE 2018 - Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, Association for Computing Machinery, Inc, Sep. 2018, pp. 259–269. doi: 10.1145/3238147.3238177.

[12] N. Kshetri, "Blockchain's roles in strengthening cybersecurity and protecting privacy," *Telecomm Policy*, vol. 41, no. 10, pp. 1027–1038, Nov. 2017, doi: 10.1016/j.telpol.2017.09.003.

[13] X. Zhang and X. Chen, "Data Security Sharing and Storage Based on a Consortium Blockchain in a Vehicular Ad-hoc Network," *IEEE Access*, vol. 7, pp. 58241–58254, 2019, doi: 10.1109/ACCESS.2018.2890736.

[14] D. Perez and B. Livshits, "Smart contract vulnerabilities: Does anyone care?," *arXiv preprint arXiv:1902.06710*, pp. 1–15, 2019.

[15] T. A. Usman, A. A. Selçuk and S. Özarslan, "An Analysis of Ethereum Smart Contract Vulnerabilities," in *2021 International Conference on Information Security and Cryptology (ISCTURKEY)*, IEEE, 2021, pp. 99–104.

[16] N. Amiet, "Blockchain vulnerabilities in practice," *Digital Threats: Research and Practice*, vol. 2, no. 2, pp. 1–7, 2021.

[17] J. Liu and Z. Liu, "A survey on security verification of blockchain smart contracts," *IEEE Access*, vol. 7, pp. 77894–77904, 2019.

[18] H. Ahmad, M. A. Ahsan and A. N. Mian, "Trends in publishing blockchain surveys: a bibliometric

perspective," *Int J Inf Secur*, vol. 22, no. 2, pp. 511–523, 2023.

[19] R. Sujeetha and C. A. S. D. Preetha, "A literature survey on smart contract testing and analysis for smart contract based blockchain application development," in *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*, IEEE, 2021, pp. 378–385.

[20] R. Pise and S. Patil, "A Deep Dive into Blockchain-based Smart Contract-specific Security Vulnerabilities," in *2022 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS)*, IEEE, 2022, pp. 1–6.

[21] Y. Kissoon and G. Bekaroo, "Detecting vulnerabilities in smart contract within blockchain: a review and comparative analysis of key approaches," in *2022 3rd International Conference on Next Generation Computing Applications (NextComp)*, IEEE, 2022, pp. 1–6.

[22] S. S. Kushwaha, S. Joshi, D. Singh, M. Kaur and H.-N. Lee, "Systematic review of security vulnerabilities in ethereum blockchain smart contract," *IEEE Access*, vol. 10, pp. 6605–6621, 2022.

[23] Y. Huang, Y. Bian, R. Li, J. L. Zhao and P. Shi, "Smart contract security: A software lifecycle perspective," *IEEE Access*, vol. 7, pp. 150184–150202, 2019.