Detecting Insider Attacks in Blockchain Networks

Oluwaseyi Ajayi and Tarek Saadawi

Department of Electrical Engineering, City University of New York, City College

Oajayi000@citymail.cuny.edu *saadawi@ccny.cuny.edu*

Abstract— **Blockchain technology has recently gained high popularity in data security, primarily to mitigate against data breach and manipulation. Since its inception in 2008, it has been applied in different areas mainly to maintain data integrity and consistency. Blockchain has been tailored to secure data due to its data immutability and distributive technology. Despite the high success rate in data security, the inability to identify compromised insider nodes is one of the significant problems encountered in blockchain architectures. A Blockchain network is made up of nodes that initiate, verify and validate transactions. If compromised, these nodes can manipulate submitted transactions, inject fake transactions, or retrieve unauthorized information that might eventually compromise the stored data's integrity and consistency. This paper proposes a novel method of detecting these compromised blockchain nodes using a server-side authentication process and thwart their activities before getting updated in the blockchain ledger. In evaluating the proposed system, we perform four common insider attacks, which fall under the following three categories: *(1)Those attacks targeting the Blockchain to bring it down. (2) the attacks that attempt to inject fake data into the database. (3) The attacks that attempt to hijack or retrieve unauthorized data.* We described how we implement the attacks and how our architecture detects them before they impact the network. Finally, we displayed the attack detection time for each attack and compared our approach with other existing methods.**

***Keywords* — B*lockchain, Cyberattack, Compromised nodes, detection time, Intrusion Detection System, Transaction, data injection.*

I. Introduction

The tremendous rise in the number of computer networks and IoT devices connected to the internet has increased their attack surface. Despite multi-level security layers, malicious intruders still find ways to subvert these protection systems and gain access to unauthorized data [1]. Further researches put forward intrusion detection systems (IDS) to identify malicious intentions in computer networks and devices connected to the internet [2,3]. Intrusion detection systems proved to help identify malicious activities; however, their single vantage point limits the ability to detect distributed or coordinated cyberattacks. As a result of the single viewpoint, some attacks can go undetected or not seen on time; hence, IDS nodes need to exchange attack information for detecting distributed attacks. A cooperative intrusion detection system (CoIDS) was proposed to improve the detecting power of single IDSs [4-6]. Companies adopted this solution because of its better performance; however, significant problems threatening the CoIDS are: (i) Data manipulation: Malicious intruders can hack the database and alter the data being exchanged even if it is not sent as plaintext. (ii) Data deletion: Stored data can be deleted from the database by a malicious insider or outsider if the activities are not monitored. (iii) Fake data injection to the database: When data manipulation is not readily achievable, a malicious intruder can inject fake data into the database if hacked. (iv) It might be challenging to guarantee the shared data's consistency due to compromised media of exchange and (v) a need to trust a third party that manages the database's activities, making the network susceptible to a man-in-the-middle attack.

Further research proposed blockchain architecture to ensure the integrity of shared data in a collaborative intrusion detection system. [7-11]. Blockchain technology was introduced in 2008 as a technology behind bitcoin to prevent double-spending in cryptocurrency[12]. Since its inception, it has been applied to different areas such as, e.g., health system [13-15], data integrity security [16], as an intrusion detection system [17 - 19], cooperative intrusion detection [7,8], and so on. Blockchain is an append-only public ledger that records all transactions that have occurred in the network. Every participant in a blockchain network is called nodes. The data in a blockchain is known as a transaction, and it is divided into blocks. Each block is dependent on the previous one (parent block). Every block stores some metadata and hash value of the last block. So, every block has a pointer to its parent block. Each transaction in the public ledger is verified by the consensus of most of the participants in the system. Once the transaction block is attached, it is impossible to mutate/erase the records [12]. Blockchain is broadly divided into two: public and private Blockchain [20]. A public blockchain is a permissionless blockchain in which all nodes do verification and validation of transactions. e.g., Bitcoin, Ethereum. On the other hand, private blockchains are permissioned blockchains that limit network participation to specific nodes. e.g., Hyperledger.

Blockchain technology proved effective in securing stored data against cyberattacks; however, its inability to identify and isolate a compromised blockchain node is one of the major problems facing the technology. Many blockchain applications in cybersecurity research focused on securing the stored data against external attacks [9, 17-19, 21,22]. However, little effort has been put into detecting internal attacks on stored data. a trust-chain among blockchain nodes was proposed in [11]. Although the solution has a good prospect in securing the integrity and consistency of shared information, it may be challenging to identify and isolate malicious nodes, especially in a private blockchain network. Identifying a malicious insider node in a blockchain network requires continuous monitoring of the node's behavior and implementing a server-side node authentication. Apart from monitoring the nodes' behavior, the transactions submitted by every node should pass through a verification process.

This paper proposed a novel architecture that can identify and thwart a compromised insider blockchain node's malicious activities on transactions by continuously examining the individual node's behavior. The proposed architecture detects the malicious insider's activities on stored attack information. We perform common insider attacks, which fall under three categories: *(1)Those attacks targeting the Blockchain to bring it down. (2) the attacks that attempt to inject fake data into the database and (3) The attacks that attempt to hijack or retrieve unauthorized data*.

The contributions of our work can be summarized as follows:

- To propose a blockchain-based architecture that can continuously monitor Blockchain nodes' activities to detect malicious nodes.
- The proposed architecture can detect insider nodes attempting to mount DoS attacks on the blockchain network.
- The architecture can detect insider nodes that attempt to inject fake data into the database or retrieve unauthorized data.
- The proposed system can verify the integrity and consistency of the submitted transaction (attack features/signature) and present it in a standard format, which encourages heterogeneous IDS node participation.
- The architecture can permanently store the verified transaction in a distributed public ledger and shares it among IDS nodes in real-time.

This paper's remainder is organized as follows: Section II discusses the background and related works on blockchain application as IDS. Section III describes the proposed architecture. Section IV presents the results, and finally, section V presents the conclusions of this paper and possible future works.

II. BACKGROUND AND RELATED WORKS

1. *Blockchain as intrusion detection systems*

The authors in [11] proposed the use of blockchain technology in detecting compromised nodes. The authors presented a trust-chain that mitigates attacks targeted at compromising intrusion detection systems. The proposed solution is to protect the integrity of the information shared among the CIDN peers, enhance their accountability, and secure their collaboration by thwarting insider attacks. A consensus protocol is proposed for CIDNs as a combination of proof-of-stake and proof-of-work protocols to enable collaborative IDS nodes to maintain a reliable and tampered-resistant trust chain. Their work focused on the theoretical aspects of security, to study a series of attacks reported in both domains (trust management and Blockchain), to fully understand the impact of various parameter choices on the proposed solution's security and the dynamics governing the trust score evolution. Although the research work has a reasonable prospect of ensuring shared information integrity, the authors failed to address how the system will identify a compromised node.

The authors in [9] present a Collaborative Blockchained Signature-based Intrusion Detection (CBSigIDS). CBSigIDS is a generic framework of collaborative blockchain signature-IDS. This framework incrementally utilizes blockchains to update a trusted signature database for different IDS nodes in a collaborative network. The experiment investigated the performance of CBSigIDS against adversarial scenarios like worm and flooding attacks in simulated collaborative intrusion detection systems or networks (CIDN). In the evaluation, they compared the results from simulated CIDN against real CIDN. The result showed that blockchain technology could enhance the robustness and effectiveness of signature-based IDSs under adversarial scenarios via building a trusted signature database. In [10], the authors proposed a SectNet, an architecture that can secure data storing, computing, and sharing in the large-scale Internet environment. The architecture aimed at more secure cyberspace with actual big data and enhanced AI with plenty of data sources. Their architecture integrates 1) blockchain-based data sharing with ownership guarantee, 2) AI-based secure computing platform, and 3) a Trusted-value exchange platform. The performance analysis evaluated the vulnerability when suffering from notorious network attacks such as the Distributed Denial of Service (DDoS) Attacks and revenue for contributors who provide Blockchain's security rules. The result showed that the SectNet significantly reduced DDoS attack's impact due to the sharing of the security rule sets by every internet user. The contributor's revenue will also increase at a higher rate if the shared security rules are of higher quality, especially after the actual market's quality effect is formed. Based on the analysis, the work is specific for DDoS attacks. Also, the authors failed to explain how stored information is verified.

In [17], the authors proposed a blockchain anomaly detection solution (BAD) that focuses on detecting attacks directed at the blockchain network. BAD prevents the insertion of a malicious transaction from spreading further in the Blockchain. BAD leverages blockchain metadata named forks to collect potentially malicious activities in the blockchain network. Their works used machine learning to train blockchain nodes to detect malicious activities. In their approach, they considered an eclipse attack (an attacker infects a node's list of IP addresses, thus forcing the victim's node list of IP addresses to be controlled by that attacker). The result analysis showed that BAD could detect and stop the spread of attack that uses bitcoin forks to spread malicious codes. However, the solution is specific to attacks directed towards the blockchain network and use bitcoin forks. In another research put forward in [18], the authors proposed collaborative IoT anomaly detection via blockchain solution (CIoTA). CIoTA uses the blockchain concept to perform distributed and collaborative anomaly detection on IoT devices. They used CIoTA to continuously trained anomaly detection models separately and then combine their wisdom to differentiate between rare benign events and malicious activities. The evaluation of the result showed that combined models could detect malware activities easily with zero false positives. The proposed solution uses a collaborative effort of IoT to detect attacks;

hence, it does not address how a compromised IoT can be identified.

The authors in [19] proposed a blockchain-based malware detection solution in mobile devices. Their work extracted installation package, permission package, and call graph package features for all known malware families for Android-based mobile devices and used them to build a feature database. Their result showed that the solution could detect and classify known malware. It also performs malice determination and malware family classification on unknown software with higher accuracy and lower time cost. The solution above is specific to host-based malware attacks on Android-based mobile devices. Hence, it will be difficult to extend it to network-based attacks.

Despite several kinds of research, the available implementations focused on protecting the shared data against outside threats. In contrast, less attention is put into detecting and isolating compromised insider threats. Our proposed architecture detects and isolates compromised insider nodes by continuously monitoring the nodes' behaviors. Apart from this, the approach also detects external threats. The above reason serves as the motivation for this work and distinguishes our work from previous related works.

III. Methodology

The proposed architecture is implemented in the laboratory on the Ethereum blockchain platform. Ethereum blockchain is an open-source blockchain-based distributed computing featuring a smart contract. A smart contract is a self-executing contract that holds the terms of agreement about a transaction, and it is written into lines of code. All participants run the code and the smart contract in a distributed, decentralized blockchain network. The smart contract automatically executes when predetermined conditions are met [23]. Although the central Ethereum platform is a public blockchain, we run it as a private network in which no public nodes can join the Blockchain. We configure the private Blockchain by building a custom genesis block and NetworkID for the Ethereum blockchain platform. Fig. 1 shows a pictorial representation of the private Blockchain.

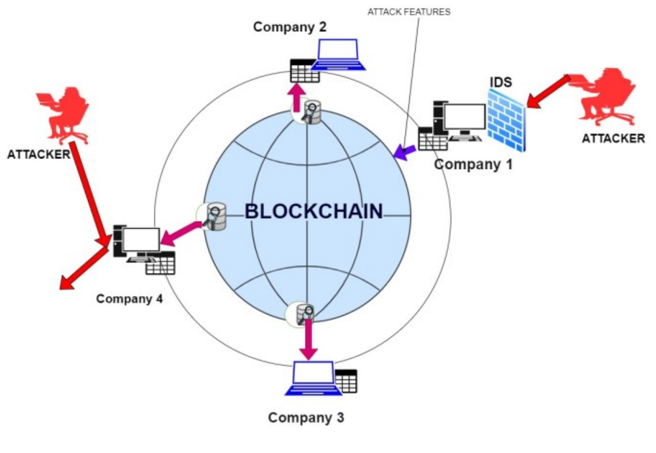

Fig. 1. The private Blockchain

The proposed architecture detects malicious activities of the blockchain nodes based on the node's heuristic analysis. The architecture's building block is divided into three stages, as shown in Fig. 2

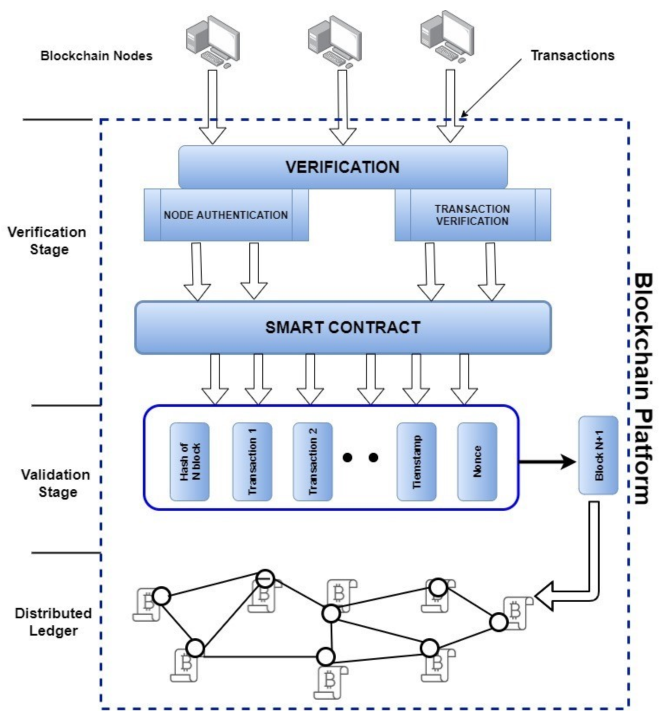

Fig. 2. Building blocks of the proposed architecture

1. *Transaction Preparations*

In our previous works [7,8,13], we described how a transaction is prepared and submitted to a blockchain network. Unlike our previous works, we implement the proposed blockchain network as a private network where all nodes can submit transactions. The submitted transaction follows an agreed-upon format written in the smart contract. As part of the action, we configure owners to submit transaction tags containing the node's transaction account alongside the transaction. The structure of the submitted transaction is shown in Table I.

Table I: Transaction format

Transaction	**Tag**
Signed Features/Signature	**Trans_acc:** *0x 8b695D0D7160aA8d95dc6ccEf6E7133F76a91De7*

2. *Verification*

i. *Node Authentication*

The smart contract handles the node authentication. The purpose of authenticating the node is to ensure that only specific nodes can submit and retrieve the information. We implement this type of policy to evaluate transaction access control in the consortium network. Authenticating a node requires that the smart contract retrieves the transaction tag and invokes a code that compares it with stored information. The information verified at this stage includes the transaction account and digital signature. The pseudocode below describes the snippet of the smart contract that handles the verification of the node. If the node authentication is successful, the algorithm invokes the transaction verification code.

Algorithm1: Node Authentication

```
Procedure: Node Authentication (Key, Information)
Inputs: Key, node Information (NI)

If (Node information is Correct) and (public key
  verifies private key):
        Return Success
        Push transaction for verification
else:
        Return fail
        Drop transaction
end if
end procedure
```

ii. Transaction verification

The transaction verification step ensures that all malicious transactions or activities on the submitted transaction by insider nodes are identified and thwarted. (i.e., it detects malicious activities and identifies the compromised node). The verification also ensures that any submitted transaction's integrity and consistency are verified before attaching it to the blockchain network. In this architecture, the smart contract behaves like a firewall that analyzes every ingress traffic to the blockchain network. We defined the maximum cost of mining a transaction since transactions are submitted in a similar format, making their mining prices almost identical. The essence of limiting the fee is to ensure that a submitted transaction is consistent with the defined transaction format and does not exhaust the resources. Any transaction costs higher than the threshold are flagged and results in a failed transaction. To minimize the influence of a compromised node in the mining process, we defined who should mine a transaction block. Here, we set a mining policy that alerts when transaction owners attempt to mine their transactions. The smart contract keeps monitoring the cost of mining each transaction and all nodes participating in mining a transaction.

Furthermore, we defined the format for submitted transactions and the maximum number of transactions per node in one second. As explained in [7,8,13], the structure of the submitted transaction is shown in Table I. We generate the key pairs for all nodes using Digital Signature Algorithm (DSA) with 512 bit-length. The public key of all nodes is written in the smart contract, while the private key is kept securely within the nodes. Every node signs the transaction with the private key, and it is verified using the public key when the transaction is submitted to the Blockchain. We also defined a transaction retrieval policy that restricts the retrieval of transactions to specific nodes. We randomly defined a privileged node list that stores the information of retriever nodes.

The purpose is to establish transaction retrieval access control in the blockchain network. A snippet from a smart contract shows algorithm 2 pseudocode describing the transaction verification process. The upper part of the code presents the submitted transaction verification algorithm. For transaction verification to be successful, the transaction must agree with the format, and the owner must not mine its transaction. Also, the cost of mining transactions must not exceed the threshold, and the number of transactions per second must not exceed the threshold value. If any of these conditions fail, the transaction is dropped, and other nodes are alerted about the malicious node's attempt. The lower part of algorithm 2 describes the condition for a transaction to be retrieved. To retrieve a transaction, the requester submits its information to the blockchain network. The smart contract invokes a privileged verification code. If successful, access is allowed; else, access denied

Algorithm2: Transaction Verification

```
Procedure: Transaction Verification (Transaction)
Inputs: Transaction

// This is invoked for a submitted transaction
If (Transaction agrees with Format) and
  (Transaction owner does not mine) and
  (Transaction cost < Max cost) and
  (Transactions/sec <= Max):
        Return Success
        Push transaction to the validation stage
else:
        Return fail
        Drop transaction
end if

// This is invoked for a retrieved transaction
 Requester invokes a privileged code
 if (Requester == a privileged node)
        verify information
        allow access
else:
        disallow access
        alert other nodes
end if
end procedure
```

3. Transaction validation

Blockchain protocol handles the validation of transactions. In this work, the blockchain platform uses both Proof-of-Work (PoW) and Proof-of-Stake (PoS). The pending transaction is built into a block, and the block is broadcasted into the blockchain network for validation. Every node receives a broadcasted block, and they work to validate the block. We set an upper bound of stake for every transaction to ensure fair competition among miners (i.e., to discourage nodes with a more significant stake from always emerge as the miner). Each block contains a unique code called hash; it also includes a hash of the previous block. Data of earlier blocks are encrypted or hashed into a series of numbers and letters. The nodes work to get the target hash to validate a block. A target hash is a number that a hashed block header must be less than or equal to for a new block to be awarded. The miners achieve this target hash by using an iterative process such as PoW, which requires consensus from all nodes. The characteristics of PoW are computationally difficult to compute and easy to verify.

The process of guessing the hash starts in the block header. It contains a block version number, a timestamp, the hash used in the previous block, the hash of the Merkle Root, the nonce, and the target hash. Successfully mining a block requires a node to be the first to guess the nonce, which is a random string of numbers and broadcast to other nodes. Other nodes verify the nonce value's correctness by appending this number to the block's hashed contents and then rehashing it. If the new hash meets the target's requirements, then the block is added to the Blockchain. The Blockchain permanently stores the transaction, and it is impossible to mutate/erase the block.

4. *Distributed Ledger*

The newly added block reflects on the ledger, which is possessed by every node in the network. The nodes receive the update of the recently added block but can not access the block's content. Smart contract handles the node's access to the block's content (algorithm 2).

IV. RESULT

The proposed architecture is implemented on an Ethereum blockchain platform. We use *Solidity v 0.7.2* implementation for smart contracts and *geth v 1.9.0* for Ethereum. For initial testing of the proof-of-concept, the private blockchain network is set up in the laboratory with five computers serving as blockchain nodes (Fig. 1) to evaluate the detection performance. We also measure the detection time of each attack and present the result in Fig.3. Table II shows the comparison of the proposed approach with other related solutions in the literature

A. *Attack launching and detection*

Here, we considered some of the everyday malicious activities of blockchain nodes. The implementation and detection of the attacks are described in this section. The attacks are divided into three categories (1)Those attacks targeting the Blockchain to bring it down (DoS), (2) the attacks that attempt to inject fake data into the database, (3) The attacks that attempt to hijack or retrieve unauthorized transaction.

1. *Denial of Service Attacks*

a) A large volume of data:

We implement a case where a compromised node sends a large amount of what appears to be legitimate standard formatted transactions in an attempt to mount a DoS attack on the blockchain network. The purpose of this attack is to exhaust all the gas prices so that when legitimate transactions are submitted, the will not be enough gas to mine the transaction. Node prepares transactions that are a massive amount of data and submit it to the blockchain network. Although other nodes are working to validate the transaction, we observed that the transactions are not mined. Notification to the owner indicates that the transaction failed due to its cost. We investigated further by manually generating the transaction address and then using it to query the Blockchain. The blockchain network did not return any transaction because no block with that transaction address resides in the network. When we check the transaction's metadata, we observed that the transaction's mining cost is greater than the smart contract threshold, hence the failed notification.

b) Multiple submitted transactions

We implement another case where a compromised blockchain node sends multiple versions of what appears to be a legitimate standard formatted transaction to mount a DoS attack on the blockchain network. The aim to overwhelm the smart contract so that some transactions can get validated without verification. The node persistently submits multiple transactions to exhaust computing resources. Although other nodes attempt the transaction's mining process, we observed that the transactions are not validated because the frequency of receiving the same or similar transaction from the same node exceeds the smart contract threshold. We persistently submit the same request from the same node, and we observed that the miners stop mining after the sender was flagged to be compromised. The smart contract automatically drops all subsequent transactions from the same authorized node.

2. *Database Injection Attempt*

a) Fake Transaction values

We implement a case where a compromised node submits what appears to be legitimate standard formatted attack features but with fake data values. The cost of each submitted transaction is within the set range in the smart contract. Generally, it is assumed that an attacker will not hold an authorized node in a compromised state for too long due to network administrators' frequent security checks. Based on this assumption, an attacker makes all efforts to get its transactions validated to the Blockchain as quickly as possible. The transaction owner attempts to mine the transaction to get it validated as fast as possible. The result showed that the transaction is not mined, although other authorized nodes are mining to validate the same transaction. The smart contract drops the transaction because the owner attempts to mine his transaction, making the transaction flagged as compromised. Information about the owner is sent to the network operator.

3. *Transaction Retrieval Attempt*

a) Unauthorized request of transaction:

We implement a situation where a node attempts to retrieve an unauthorized transaction. We implement this malicious activity to demonstrate how the proposed architecture can control transaction retrieval access. The node queries the Blockchain for the content of a newly added block in which it is not privileged to download. The result showed that no information was returned because the node is not privileged to retrieve the data. The smart contract handles each transaction's access control based on the access level submitted with the transaction. We investigate further by querying the Blockchain using a node that has access to retrieve the data. The node successfully downloads the block's content from the Blockchain.

B. Performances Analysis

a. Detection time

We evaluated the detection time of attacks. The detection is defined as the time it takes for the architecture to notify about the transaction's failure. This time is measured from when the transaction is submitted to the time the notification is received. Both the submission and notification are timestamped and recorded for each attack. The detection time is the difference between these two times. Fig.3 shows the difference in the detection time for the attacks under consideration. We observed that the time taken to detect a DoS attack with multiple transactions is the highest. The higher time is because we delayed the smart contract to count the number of transactions per second before sending the failure notification. Finally, we carry out a comparative study of our approach with solutions from other similar works (Table II). The result shows a clear distinction between our work and other related works.

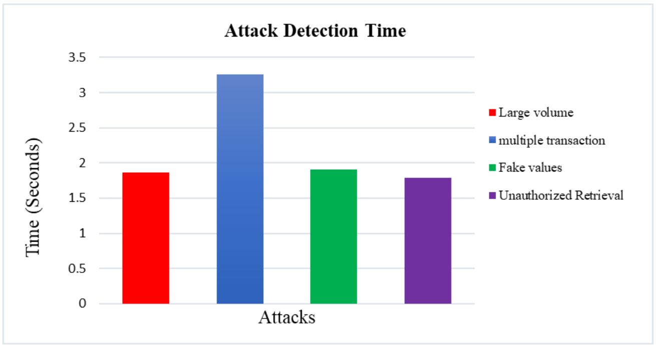

Fig. 3. Detection Time of attacks under consideration

Table II: Performance comparison with other approaches

Properties	**Trust-chain [11]**	**CBSigIDS [9]**	**SectNet [10]**	**BAD [17]**	**Our Method**
Data Sharing	✓	✓	✓	x	✓
Blockchain	✓	✓	✓	✓	✓
Detect External Node	x	✓	✓	x	✓
Detect Insider Node	✓	x	x	x	✓
Compatible with different IDS	x	✓	x	x	✓
Smart Contract verification	x	x	x	x	✓

V. CONCLUSION

In this paper, we proposed a novel architecture that can identify and thwarts the malicious activities of a compromised insider blockchain node by continuously examining the individual node's behavior. In the proof-of-concept, we set up a private blockchain network in the lab and describe how the smart contract performs the authentication and detection of malicious nodes. We evaluated the performance under the following three categories of attacks: *(a)Those attacks targeting the Blockchain to bring it down (b) the attacks that attempt to inject fake data into the database (c) The attacks that attempt to hijack or retrieve unauthorized data.* We further evaluate the performance of the architecture by observing the attack detection time. The result shows that the proposed architecture has a reasonable prospect of detecting and isolating typical insider malicious activities.

REFERENCES

[1] S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas, "Modeling intrusion detection system using hybrid intelligent systems," Journal of network and computer applications, vol. 30, no. 1, pp. 114–132, 2007.

[2] O. Igbe, O. Ajayi, and T. Saadawi, "Denial of Service Attack Detection using Dendritic Cell Algorithm" 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON 2017) Oct 19th – 21st 2017, Columbia University, New York, USA.

[3] O. Igbe, O. Ajayi, and T. Saadawi, "Detecting Denial of Service attacks using a combination of Dendritic Cell Algorithm(DCA) and Negative Selection Algorithm(NSA)" 2nd International Conference on Smart Cloud (Smart Cloud 2017) Nov 3rd-5th, 2017, New York, USA.

[4] Y. L. Dong, J. Qian, M. L. Shi, "A cooperative intrusion detection system based on autonomous agents," IEEE CCECE 2003, Vol. 2, pp. 861– 863, 2003.

[5] C. C. Lo, C. Huang, J. Ku, A cooperative intrusion detection system framework for cloud computing networks, in: In: Proceedings of the 2010 39th International Conference on Parallel Processing Workshops,ICPPW '10, 2010, pp. 280-284.

[6] Y.-S. Wu, B. Foo, Y. Mei, and S. Bagchi, "Collaborative intrusion detection system (CIDS): A framework for accurate and efficient IDS," in Proc. Annu. Comput. Secur. Appl. Conf. (ACSAC), Dec. 2003, pp. 234–244.

[7] O. Ajayi, M. Cherian and T. Saadawi, "Secured Cyber-Attack Signatures Distribution using Blockchain Technology." 2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), New York, NY, USA, 2019, pp. 482-488.

[8] O. Ajayi and T. Saadawi, "Blockchain-Based Architecture for Secured Cyber-Attack Features Exchange," 2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), New York, NY, USA, 2020, pp. 100-107, doi: 10.1109/CSCloud-EdgeCom49738.2020.00025.

[9] S. Tug, W. Meng and Y. Wang, "CBSigIDS: Towards Collaborative Blockchained Signature-Based Intrusion Detection," 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 2018, pp. 1228-1235, doi: 10.1109/Cybermatics_2018.2018.00217.

[10] K. Wang, J. Dong, Y. Wang and H. Yin, "Securing Data With Blockchain and AI," in IEEE Access, vol. 7, pp. 77981-77989, 2019, doi: 10.1109/ACCESS.2019.2921555.

[11] N. Kolokotronis, S. Brotsis, G. Germanos, C. Vassilakis and S. Shiaeles, "On Blockchain Architectures for Trust-Based Collaborative Intrusion Detection," 2019 IEEE World Congress on Services

(SERVICES), Milan, Italy, 2019, pp. 21-28, doi: 10.1109/SERVICES.2019.00019.

[12] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system.", 2008

[13] O. Ajayi, M. Abouali and T. Saadawi, "Secure Architecture for Inter-Healthcare Electronic Health Records Exchange," 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Vancouver, BC, Canada, 2020, pp. 1-6, doi: 10.1109/IEMTRONICS51293.2020.9216336.

[14] T. Ahram, A. Sargolzaei, S. Sargolzaei, J. Daniels, and B. Amaba. "Blockchain Technology Innovation". 2017 IEEE Technology & Engineering Management Conference (TEMSCON), 2017

[15] Liang, X.; Zhao, J.; Shetty, S.; Liu, J.; Li, D. Integrating Blockchain for data sharing and collaboration in mobile healthcare applications. In Proceedings of the 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Montreal, QC, Canada, 8–13 October 2017.

[16] Zikratov, I., Kuzmin, A., Akimenko, V., Niculichev, V., Yalansky, L.: Ensuring data integrity using Blockchain technology. In: Proceeding of the 20th Conference of fruct Association ISSN 2305-7254 IEEE (2017)

[17] M Signorini and M Pontecorvi, W Kanoun, and R Di Pietro, "BAD: a Blockchain Anomaly Detection solution" arXiv:1807.03833v2, [cs. CR] 12 jul 2018

[18] T. Golomb, Y. Mirsky and Y. Elovici "CIoTA: Collaborative IoT Anomaly Detection via Blockchain" arXiv:1803.03807v2, [cs. CY] 09 Apr 2018

[19] Gu, J, B Sun, X Du, J Wang, Y Zhuang and Z Wang (2018). Consortium blockchain-based malware detection in mobile devices. IEEE Access, 6, 12118–12128

[20] Abdullah, N., Hakansson, A., & Moradian, E. (2017). Blockchain based approach to enhance big data authentication in distributed environment. In Ubiquitous and future networks (icufn), 2017 ninth international conference on (pp. 887–892).

[21] M. Kumar and A. K. Singh, "Distributed Intrusion Detection System using Blockchain and Cloud Computing Infrastructure," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), Tirunelveli, India, 2020, pp. 248-252, doi: 10.1109/ICOEI48184.2020.9142954.

[22] B. Jia and Y. Liang, "Anti-D chain: A lightweight DDoS attack detection scheme based on heterogeneous ensemble learning in blockchain," in China Communications, vol. 17, no. 9, pp. 11-24, Sept. 2020, doi: 10.23919/JCC.2020.09.002.

[23] Ingo Weber, Vincent Gramoli, Mark Staples, Alex Ponomarev, Ralph Holz, An Binh Tran, and Paul Rimba. 2017. On Availability for Blockchain-Based Systems. In SRDS'17: IEEE International Symposium on Reliable Distributed Systems