Identification of Compromised Nodes in Collaborative Intrusion Detection Systems for Large Scale Networks Due to Insider Attacks

by

Chandralekha Yenugunti

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2020 by the
Graduate Supervisory Committee:

Stephen S. Yau, Chair
Yezhou Yang
Jia Zou

ARIZONA STATE UNIVERSITY

May 2020

ABSTRACT

Large organizations have multiple networks that are subject to attacks, which can be detected by continuous monitoring and analyzing the network traffic by Intrusion Detection Systems. Collaborative Intrusion Detection Systems (CIDS) are used for efficient detection of distributed attacks by having a global view of the traffic events in large networks. However, CIDS are vulnerable to internal attacks, and these internal attacks decrease the mutual trust among the nodes in CIDS required for sharing of critical and sensitive alert data in CIDS. Without the data sharing, the nodes of CIDS cannot collaborate efficiently to form a comprehensive view of events in the networks monitored to detect distributed attacks. The compromised nodes will further decrease the accuracy of CIDS by generating false positives and false negatives of the traffic event classifications. In this thesis, an approach based on a trust score system is presented to detect and suspend the compromised nodes in CIDS to improve the trust among the nodes for efficient collaboration. This trust score-based approach is implemented as a consensus model on a private blockchain because private blockchain has the features to address the accountability, integrity and privacy requirements of CIDS. In this approach, the trust scores of malicious nodes are decreased with every reported false negative or false positive of the traffic event classifications. When the trust scores of any node falls below a threshold, the node is identified as compromised and suspended. The approach is evaluated for the accuracy of identifying malicious nodes in CIDS.

DEDICATION

I dedicate this thesis to my parents Sreekanth Yenugunti and Komala Yenugunti. I am thankful for their unconditional love, blessings, and constant faith in me. I will always strive to make them proud.

ACKNOWLEDGMENTS

I would like to thank my advisor Professor Stephen Sik-Sang Yau, for his guidance in my research and the opportunity to work with him. He was extremely helpful, patient with my thesis study, and is a constant source of inspiration to work hard and strive for perfection. I would also like to thank my graduate supervisory committee members, Professor Yezhou Yang and Professor Jia Zou.

I would also like to express my gratitude to my laboratory colleagues Jinal Patel, Suli Adeniye, Alex Nou and Brandon Anderson for their valuable inputs during the countless research meetings and discussions.

TABLE OF CONTENTS

Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

There are about 17 billion electronic devices and about 7 billion IoT devices worldwide in use, according to statistics in 2018 [1]. This number is expected to cross 20 billion by 2025. With the growing size of internet and connectivity, the number of cyberattacks and the sophistication of the attacks is ever increasing. In the current Cyberspace, there are multitude types of devices such as smartphones, tablets, laptops, Internet of things that are always connected to the internet, making them vulnerable to attacks originating from anywhere in the world. There are diverse types of networks such as Personal Area Networks, Wide Area Networks, Campus Networks, Ad Hoc Networks, Enterprise Networks, etc, which need state of the art defenses. With the world taking giant leaps towards complete digitalization, everyone is gaining access to network and computation resources as never before. The attackers are able to have access to hacking resources, computation resources, and collude to launch attacks ranging from newbie attacks to large scale breaches compromising data and causing damages in the order of millions of dollars. The most recent example for one such breach is the Capital One data breach. The breach was detected in July 2019, as published by Capital One. The breach compromised the critical data of about 106 million individuals [2]. Although the breach happened in late March, it was not detected until July. It is in situations like these, the importance of Intrusion Detection Systems arises. They are one of the first lines of defense against security attacks and breaches. However, stand-alone Intrusion Detection Systems can be by-passed and erasing the traffic logs for removing the tracks of the attacker [3].

This gives rise to the need for Collaborative Intrusion Detection Systems, [4] which can detect large scale distributed attacks and are resilient to attacks against the Intrusion Detection System itself. While the Collaborative Intrusion Detection Systems are extremely efficient in detecting distributed attacks, these are prone to internal attacks [5]. While there are external attacks on Collaborative Intrusion Detection Systems, internal attacks are the primary focus in this thesis. This thesis elaborates on the challenges of insider attacks and data sharing in Collaborative Intrusion Detection Systems internal and an approach based on blockchain is presented to address these challenges. A thorough study of Collaborative Intrusion Detection Systems, their challenges was performed. After carefully analyzing various potential solutions, blockchain [6] was chosen as its features showed promising solutions. Further, blockchain was rigorously studied to learn how to apply it to the approach. Later, the approach was implemented to study the effectiveness of the approach and a detailed analysis is given to show the results of using a blockchain-based approach for Collaborative Intrusion Detection Systems.

This thesis is organized into 7 chapters. The first chapter gives the introduction of the research and the overview of this document. The second chapter provides the background and context required to understand the research. In the third chapter, the problem statement, the current approaches and their shortcomings are discussed. The fourth chapter presents our overall approach to address the problem. The approach contains the framework and the blockchain-based solution. The fifth chapter discusses a detailed explanation of our approach using an example. The sixth chapter consists of an evaluation of approach and analysis of the results of simulation of approach. The seventh chapter is the conclusion and gives a brief overview of future work.

CHAPTER 2

BACKGROUND

2.1 Intrusion Detection Systems

Intrusion Detection Systems are software applications that are deployed to monitor computer networks, computer systems. IDS gained importance in the ‘90s, and significant research is ongoing since then [7]. IDS are configured to classify the traffic logs and events into two categories, malicious and benign events. Malicious events include attempts to detect, exploit vulnerabilities and attacks. The malicious events detected by IDS include events ranging from reconnaissance attacks [8, 9] such as OS fingerprinting on computer systems, scanning attacks [9] such as port scanning to detect open ports, detecting established malicious connections, denial service attacks on networks. IDS are used to identify both active and passive attacks by analyzing the logs 24x7. IDS are of vital importance in detecting the exact sequence of steps which lead to the attack and identify the time, source, vulnerabilities, and the damage caused by the attack. IDS have primarily two different kinds of classifications, based on the scope of monitoring and based on the method of detection. IDS can be classified into Host-based Intrusion Detection Systems (HIDS) and Network-based Intrusion Detection Systems (NIDS) [7]. IDS are also classified into Signature-based Intrusion Detection System and Anomaly-based Intrusion Detection Systems.

Host-based Intrusion Detection Systems: HIDS are deployed on a single host and monitor the data and the events on this single system. HIDS monitors every activity ranging from

operating system-level events, application events, and network events. HIDS looks for malicious activities on the host such as privilege escalation, applications requesting data that they do not have access to, network intrusions and many such events.

Network-based Intrusion Detection System: NIDS is used to monitor the entire network or subnets and typically deployed at the point where all traffic enters and leaves the data. The network traffic data is be processed in real-time as well as offline to generate real-time alerts. NIDS monitor and analyze the network for intrusions ranging from scanning attacks to large scale attacks such as distributed denial of service attacks. NIDS can use different detection methods as mentioned above [7].

Signature-based IDS: Signature-based IDS monitors every packet of network traffic and compare them against a known database of attacks and threats. Each of the events in the traffic is checked against the patterns/signatures of these known attacks and threats. Signature-based IDS can detect any known attacks but are not very effective against novel attacks or zero-day attacks [7]. Although they are very fast and efficient to detect known attack patterns, Signature-based IDS can have a huge computational overhead as they have to perform analysis on each packet for detection of any signature from the database. Signature-based IDS can fail, even if there is a slight modification of the known attack and can be easily by-passed [7].

Anomaly-based IDS: Anomaly-based IDS [10] are deployed to detect novel attacks that are not seen before. An anomaly-based IDS builds a profile of regular or expected behavior

from the traffic events and analyzes new incoming traffic to detect malicious events. Anomaly-based IDS are extensively used for network intrusion detections. Various machine learning techniques such as unsupervised machine learning are used for building the profile, and detecting anomalies are used, when there are no labeled traffic events that can be used for training the models used by IDS. Supervised machine learning techniques are used when there is availability of classifications of the events and attacks. Other techniques such as neural networks, data mining, etc are used for improving anomaly-based detections in IDS. While anomaly-based IDS is very efficient in detecting novel attacks and intrusions, it can lead to reporting a lot of false alarms if the initial profile building is not monitored carefully [3].

Hybrid IDS are used where they combine the detection mechanisms of signature-based and anomaly-based IDS. These IDS can also monitor the host systems as well as networks [11]. While the IDS are very efficient in detecting intrusion, they are vulnerable to attacks on IDS itself. An attacker can also map the network, and gain the information about the location, type of IDS and the detection mechanisms, and use this information to by-pass the IDS. Moreover, stand-alone IDS can be overwhelmed by denial of service attacks, after which the attacker can continue to perform malicious activity while being undetected [4].

2.2 Collaborative Intrusion Detection Systems

Collaborative Intrusion Detection Systems (CIDS) are a group of Intrusion Detection Systems working together collectively by monitoring and analyzing traffic

events to detect any intrusions and malicious activity [4]. A stand-alone IDS cannot obtain a correlation between distributed attacks happening on various parts of a network. However, CIDS offer a holistic view of the events in a network or computer systems for detecting any suspicious activities. Unlike the traditional stand-alone IDS systems which are vulnerable to attacks on the IDS itself, CIDS is very resilient against such attacks because of multiple monitors working together.

Each individual unit part of CIDS is called a node in further reference. Each node can be a monitoring unit or an analyzing unit or both. A monitoring unit records and logs the network traffic. Based on the architecture, CIDS are classified into three types: Centralized, Hierarchical, and Distributed [4].

Centralized CIDS: In the CIDS with centralized architecture, the monitoring nodes log the recorded traffic data and send it to a centralized analyzing unit to detect any intrusions. Centralized CIDS is prone to Single Point of Failure (SPoF) and is not scalable because of a single central analyzing unit [12].

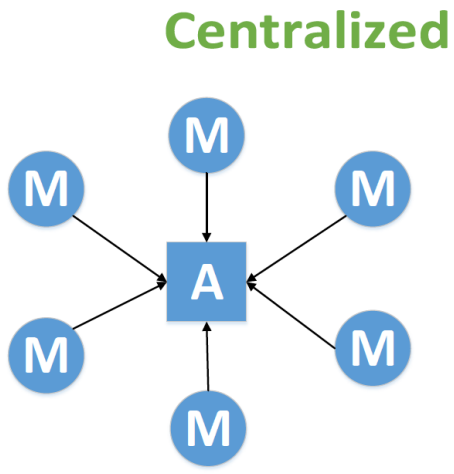

Figure 1: Architecture of Centralized CIDS [4]

Decentralized/Hierarchical CIDS: In the CIDS with hierarchical architecture, the monitoring nodes send the recorded traffic data to a local analyzing unit, which in turn sends the relevant events to another analyzing unit to detect intrusions. This forms a hierarchy in the architecture of nodes. Due to the decentralized CIDS, this CIDS is less prone to SPoF and more scalable. However, hierarchical CIDS is still prone to SPoF and not entirely scalable.

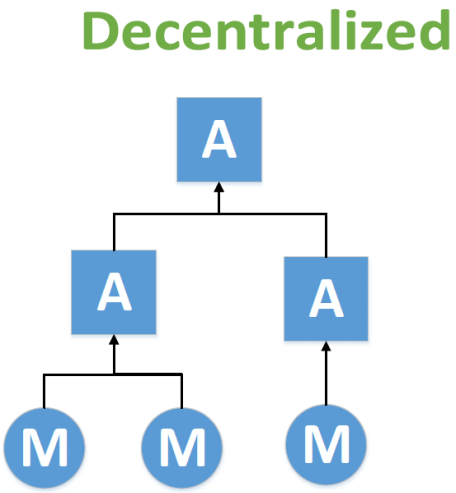

Figure 2: Architecture of Decentralized CIDS [4]

Distributed CIDS: In this CIDS with distributed architecture, the nodes are in a distributed peer-to-peer fashion, where every node is both analyzing and monitoring units. Due to the decentralized and distributed nature, this CIDS is not prone to SPoF attacks and is highly scalable. This thesis emphasizes on the approach to address the challenges of CIDS with distributed architecture.

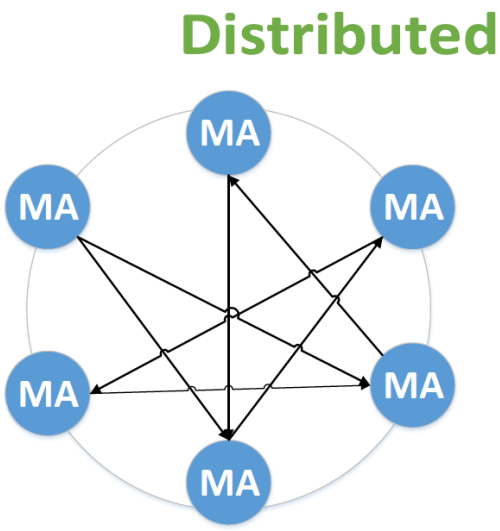

Figure 3: Architecture of Distributed CIDS [4]

The following are the requirements of a CIDS: [4]

Accuracy: A CIDS should minimize false positives and false negatives of the classifications to increase the overall accuracy.

Accountability: Every participating node should be held accountable for every action while analyzing, monitoring the events, sharing, and exchanging information between the nodes of CIDS.

Integrity: The recorded traffic events, the intrusions detected should maintain integrity for auditable purposes.

Overhead: The real-time analysis of the intrusions requires that the communication overhead between the nodes should be minimized. The computation overhead should be decreased to increase the response time to detect intrusions

Privacy: The data in the traffic logs and the alert data should be shared in a secure way. The data exchanges as part of the logs between the nodes must be protected as it may contain sensitive and critical information

Scalability: A CIDS should be able to scale to monitor and analyze the traffic events in real-time while not being a bottle for SPoF attacks. The capability of CIDS should increase proportionately with the resources of CIDS

2.3 Blockchain

Blockchain is one of the technologies with the hype of being a disruptive technology. Blockchain was first introduced with Bitcoin's [13] inception in 2009. Although primarily used as the technology behind cryptocurrencies, blockchain is being used and incorporated in various applications. In layman terms, a blockchain is a distributed ledger of digital transactions. It is decentralized, removing the need for a central authority or third parties. Trust in blockchain is established by consensus (Merriam-Webster dictionary's definition of consensus is a general agreement: unanimity) of all participants in blockchain to do something. Blockchain has properties of being decentralized, distributed, secure, and immutable. It addresses the problems faced by centralized technologies such as single-point-of-failure. Before jumping on to the technical aspects, application, and challenges of blockchain, it is important to remember that blockchain should be thoroughly understood before using it in any application and not fall for the hype that blockchain is the panacea to all modern digital problems. Blockchain uses several cryptographic mechanisms in its core features. Three of these important concepts are hash functions, asymmetric key cryptography, and digital signatures [14].

Hash functions: Hash function is a function that takes an input of any size and maps it to an output of fixed size, usually called a message digest [14]. One example of a hash function is SHA-256, which takes an input of any size and gives an output of 256 bits. Even the smallest change in the input gives different output. The hash functions used in the current blockchains are SHA-256, Scrypt.

Asymmetric Key Cryptography: Blockchain uses public-key cryptography for digital addresses corresponding to transactions [14]. Asymmetric key cryptography uses a pair of keys, public and private keys. The private keys are known only to the owner, and the corresponding public keys are published online, accessible for everyone.

Digital Signatures: Public and private keys are used as digital signatures in Blockchain. For example, Alice will sign her transactions with her private key, which can be verified by anyone using Alice's public key.

Blockchain network: Any electronic device with resources that participate in a blockchain network is called a node. Every node in the blockchain network has the responsibility of storing the entire copy or part of blockchain [14]. Full nodes are the nodes that store the entire copy of blockchain and are available online all the time.

Blockchain: Blockchain is a distributed ledger, which holds the records of a list of transactions in blocks which are linked together by cryptographic hash functions. In essence, blockchain is like a linked list, where the current block is connected to the previous block [6].

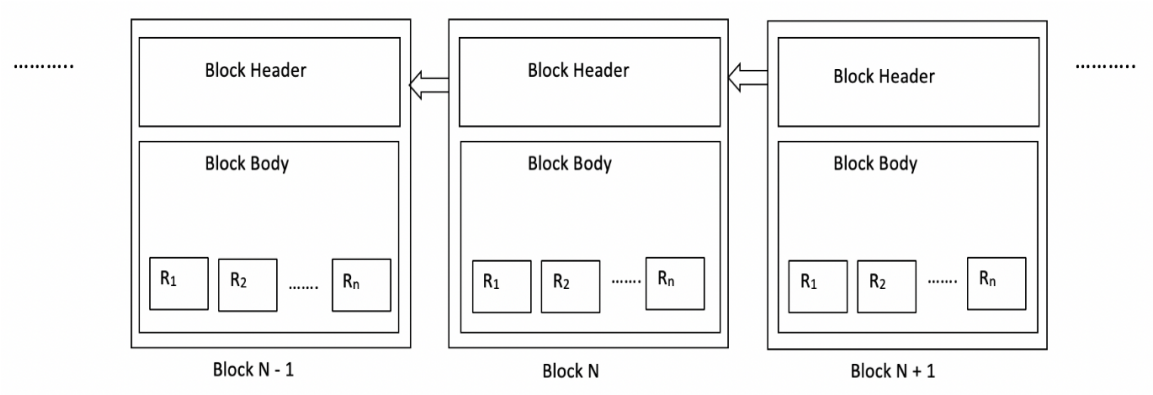

Figure 4: A Snapshot of Blocks Connected in Blockchain

Blockchain Structure: Each block in the blockchain has a header and a body.

Block header: A header consists of the following fields:

Block version: The version of the blockchain to follow.

Merkle tree root hash: Merkle trees [15] are data structures that store the hash of the data in the leaves. Each internal node is a hash of its child nodes. The nodes are created bottom-up until the root of the tree is created. This root hash of the Merkle tree is stored in the Block header. In blockchain, the hashes of the transactions are stored in the leaves of the Merkle tree and the tree is created. If any transaction is tampered, the corresponding hash of the transaction in the leaf of tree changes. This change of hash propagates to the root of the tree. The new root hash will be different from the hash from the untampered data. This changes the hash of the header, which will no longer match with the hash of this header stored in the next block. Since the root hash is obtained from the data in the block body, the integrity of the block data can be verified by checking the Merkle root hash in the header.

Nonce: A nonce (Number used only ONCE) is a cryptographic random number [14]. A Nonce is an optional field used in the proof of work consensus-based blockchains. The Nonce is the only field in the block header, which can be changed. Proof of work consensus models leverages this to calculate the hash of the header in such a way that it is lesser than a specified target hash value, by changing the nonce continuously until the hash requirements are met.

Previous block hash: The hash of the previous block's header is stored in the header of the current block. This acts as the "***cryptographic link***" to the previous block [6]. If any field in the previous block's header or body is changed, the new hash of the previous block will not match the hash of the previous block stored in the current block's header. The attacker should change the "previous block hash" field stored in all blocks after the block, which has been tampered to make the tampering undetected. This requires the attacker to create a chain longer than the current chain of the blockchain, which is resource-intensive and almost impossible.

Genesis Block: This is the first block in the blockchain, in which the previous block hash is set the zero. This block is created with all the configurations of the blockchain, such as the consensus model to follow, the hash functions to be used, the block creation rate, and other custom features.

Block body: This consists of data in forms of transaction records, health records, IoT data depending on the application.

Proof of Work (PoW): Before a new block is added to the blockchain, each node attempts to calculate the hash value of the block header which is required to be smaller than a target value [16]. Thus, nonce is changed continuously until the hash with the specified requirements is calculated. This method of calculating hash is called “mining”, and the nodes computing the hash are called “miners” in bitcoin [13]. These calculations for the hash consume a significant amount of computational resources and energy. Since calculating the nonce is a random process, it is unpredictable which node will generate the hash value. The first node to successfully find the hash will broadcast the value to all the nodes in the network. When all the nodes in the network verify and accept the hash, the new block with the hash is appended to the blockchain by the node and the node which calculated the hash is rewarded. This provides nodes an incentive to participate in the consensus and stay honest because tampering with the blocks decreases the trust and thus undermining the value of the incentive in the network. Cryptocurrencies use blockchains with PoW consensus. Examples of cryptocurrency PoW are Bitcoin, Ethereum [17], etc.

Proof of Stake: Proof of Stake (PoS) [14, 16] algorithms were introduced as an alternative to the high resource consuming proof of work algorithms. It is based on what is at “stake” by the participating node, for example, the cryptocurrency held by the node. The next node that can append a block to the blockchain is selected with a probability proportional to what is at stake by the node. Although PoS is energy efficient as compared to PoW, it is easier to attack PoS based blockchain compared to PoW based blockchain.

Delegated Proof of Stake: Delegated Proof of Stake (DPoS) [16] consensus algorithms are based on the concept of democracy. The blockchains using DPoS have nodes called "witnesses" and "delegates". The users elect witnesses and delegate by voting tokens. However, any user can participate in the voting process, unlike in PoW, where only nodes with enough computational resources can influence the consensus decision. The selected witnesses are responsible for block creation and validation and are awarded, which is the incentive to maintain the reputation of honest behavior. Malicious behavior by witnesses is prevented by active voting, as the reputation of the witness is at stake. DPoS is faster compared to PoW and PoS, and energy-efficient and offers protection from double-spending attacks. However, DPoS suffers from the same problems as voting in real life, some users may not participate in voting and requires everyone to participate.

Proof of Authority: PoA consensus algorithms [14] uses the identity of the user as the stake. All the validations and transactions by the node participating in the consensus are stored on the blockchain. Having the identity at stake ensures that nodes engage in honest behavior, as loss of reputation can be damaging. While PoA has the advantages of scalability and low resource requirement, having a few nodes participating in consensus can lead to centralization and ability for these nodes for censorship. The users ready to face the consequences of reputation damage may put the whole blockchain at stake.

Practical Byzantine Fault Tolerance (PBFT): In this consensus model, at least two-thirds of nodes should agree for a block to be added to the blockchain. This model work, even when up to one-thirds of nodes are malicious. This method is prominently used in

Hyperledger consensus for adding blocks. This model has lesser computational overhead and latency compared to other consensus models and offers high throughput [17].

Characteristics of Blockchain

Immutable: Once a block with approved transactions is created in blockchain, it is almost impossible to tamper with the data due to heavy resource requirements to do this, while not being noticed by other nodes in the networks, as everyone has their own copy of blockchain and can verify comparing their own copy [16].

Decentralization: When transactions are processed, and a new block has to be created, there should be a consensus with everyone in the network. This removes the need for approval from a centralized authority or any third parties as compared to a traditional centralized system [14].

Anonymity: The transactions of records in blockchain take place with the use of digital signatures, using public and private keys of the nodes, thus keeping the identity of the users anonymous [16].

Transparency: Every node on the blockchain has access to all the records, thus making blockchain transparent and verifiable at any point of time.

Auditability: Every block in the blockchain is appended with a timestamp recorded in the header.

Classification of Blockchains

Blockchains can be classified into three categories [16]: Public, Private, and Consortium Blockchains. Each category has its own set of advantages and disadvantages.

Public blockchains: Anyone with required computational and storage resources can participate in public blockchains. Every participating node in blockchain will have read, write, and access to the blockchain ledger and can participate in the consensus. Bitcoin and Ethereum are examples of public blockchains.

Private blockchains: These are owned by a single organization and these are regarded as partially centralized as the permissions are controlled by the organization. Only authorized users by the organization will have access to the blockchain and can participate in the consensus process. Ripple is an example of private blockchain.

Consortium Blockchains: In these blockchains, the consensus process is limited to a pre-selected set of nodes. These blockchains can be considered as a hybrid of public and private blockchains. These are widely used in financial sectors between multiple organizations. Hyperledger by IBM is an example of a consortium blockchain.

Challenges of using blockchain:

51% attack: In blockchains using proof of work consensus mechanisms, the probability of mining a block depends on the CPU cycles spent calculating the hash. Because of this, miners share the processing power to mine the blocks and split the reward based on their contribution of the resources. These groups of miners are called “mining pools”. If any

mining pool can control 51% of the computing power, the group can undermine the security of blockchain [18]. Since the group holds the majority of the computing power, the group can find the nonce faster than any other miners and mining pools, which can lead to a double-spending attack, modifying the blockchain. Blockchain is secure under the assumption that at least 51% of the miners are honest.

Scalability: Blockchain requires that all the participating nodes in the blockchain network have a complete copy of the entire blockchain all the time. This requires a huge storage overhead [16]. As the blockchain network continues the transactions, the nodes should keep up to date by downloading new blocks, thus the storage resource requirements keep growing. Nodes without enough resources cannot actively participate in the blockchain network.

CHAPTER 3

PROBLEM STATEMENT

Intrusion Detection Systems (IDS) are widely used to prevent attacks by analyzing the traffic and generating appropriate alerts. However, a single, isolated IDS does not fare well with distributed attacks and may fail in recognizing the attacks. Far worse, the IDS itself may be attacked and compromised, after which there is no way to log the activities of an unauthorized user in the compromised network. For this purpose, CIDS are used. CIDS use multiple IDS nodes spread across the network to analyze the traffic distributed throughout the network to detect any potential distributed attacks by having a holistic view of the network.

However, CIDS still have two major challenges remaining, which have to be addressed. CIDS have the challenges of trust management among the nodes and data sharing problems [19]. CIDS are prone to external attacks and internal attacks. The internal attacks occur due to a malicious insider, which compromises the entire CIDS. These attacks are difficult to address when the compromised insider node is communicating with an external malicious entity. The compromised node will then proceed to suppress valid alerts corresponding to an external attack, generate and forward fake alerts which decrease the accuracy of CIDS completely compromising the validity of CIDS. The malicious insiders lead to a decrease of trust among the nodes of CIDS [19]. The decreased trust among the nodes leads to data sharing problems when the nodes refuse to share traffic logs due to privacy concerns. Without efficient data sharing between the nodes, the CIDS ability to have a holistic view of traffic is affected and the CIDS is no longer reliable as an IDS.

Various approaches are presented to address the above challenges such as trust-aware CIDS, trust management [20], adaptive trust score systems, peer-to-peer overlay systems [21] and Dirichlet-based trust management systems [22] and blockchain-based solutions [23]. However, trust management is often handled by reputation systems, which themselves are prone to betrayal attacks [20]. An attacker who can gain access to CIDS can tamper the trust scores stored affecting the reliability of the system. Trust management often needs a trusted central authority in charge of assigning and managing the trust scores. A compromised central authority can further compromise the entire trust system and CIDS. The blockchain-based solutions present an approach, without implementation or a thorough analysis of the approach and its performance. Additionally, there is also a need to protect the privacy of the sensitive and critical information exchanged between the CIDS nodes, while also minimizing the communication overhead. While there are other challenges, such as external attacks on the CIDS, it is important to address the internal attacks to maintain resilience against the external attacks [24]. Thus, this research focuses on addressing various internal attacks - newcomer attacks, sybil attacks, and betrayal attacks and data sharing problems. A blockchain-based approach is presented which is implemented as a blockchain consensus to address the internal attacks. Addressing the internal attacks leads to an increased level of trust among nodes of CIDS, which in turn solves the data sharing problem of CIDS.

CHAPTER 4

OVERALL APPROACH

In the approach, a trust score [25] based approach is implemented on private blockchain. The next section discusses the framework design to implement the approach.

4.1 Framework Design

The framework consists of CIDS nodes and a private blockchain. In the framework, a CIDS node is initialized to be a part of the private blockchain network. Every CIDS node is also a blockchain node. A trust score-based approach will be implemented on a private blockchain, where the trust scores are stored on blockchain. The following are the components and features of the framework:

Blockchain: A private blockchain is used to implement the approach discussed in the next subsection. The Blockchain provides the following mechanisms in the framework: Identity verification of the nodes joining the CIDS as well as blockchain, trust computation through consensus, and communication. The trust scores of every node and validated traffic alerts by every CIDS node are stored on the blockchain. The approach will be implemented on the smart contract [26] of the private blockchain.

CIDS: The CIDS nodes analyze the traffic to detect and identify traffic alerts. The alerts classified are shared with other nodes in CIDS through blockchain. The alerts that are validated through the approach are stored on the blockchain. Every CIDS node will have a

copy of blockchain consisting of alerts identified by every node, which will help in efficient sharing and obtaining a comprehensive view of the attacks and alerts being detected in the network.

Identity verification: All the nodes that are part of CIDS are required to be a part of the blockchain and the consensus of blockchain. Before joining the private blockchain, the nodes are required to verify the identity, after which, the nodes will be given a public, private key pair, binding them to the node's identity. Only verified nodes can participate in the consensus of blockchain.

Trust Computation through consensus: The overall approach discussed in the next section is implemented on the blockchain through consensus. The blockchain consensus is used to determine the validity of attacks and alerts and adjust the trust scores based on that. The consensus is reached among the nodes by evaluating the credibility of each alert received from a node by verifying the traffic logs using the IDS component of each node of CIDS and blockchain. The incentive for the nodes to participate in the consensus is to increase the trust score and be an active node in the framework. The transactions validated through consensus are grouped together to form the next block. The next node to generate the block is selected using the PBFT consensus system similar in Hyperledger [17, 27], from the group of nodes with trust scores above the threshold.

4.2 Trust Score Approach to Identify Compromised Nodes

In the approach, a trust-score based approach is implemented using a private blockchain. The motivation in this approach is to force the malicious nodes to act honestly. The assumption of this trust-based approach is that honest nodes always act honestly, with or without an incentive. The trust scores of every node are stored on the blockchain, thus every node has a copy of blockchain ledger and is aware of the trust score of every other node. The trust scores of the nodes are in the range of [1 - 100] [25]. When a new node is joining CIDS, its identity is verified before adding it to both CIDS and the private blockchain of the framework. The default trust score of a new node is 50. This approach requires all the nodes to participate in the consensus all the time.

The CIDS nodes continuously monitor and analyze traffic events to generate the classification of the events. In this approach, the CIDS classifications of an event can be in of the two categories, an alert or benign event. When events that are classified as alerts are detected at any node, the node broadcasts all such alerts, to obtain consensus before adding the alert to the blockchain. All the validated alerts generated by every node are added to the blockchain. Every CIDS node has the blockchain copy through which the CIDS can build a global view of the alerts happening in the entire network being monitored by CIDS. The following sequence of steps gives greater detail of the approach to address the internal attacks.

Terms used in the approach for consensus:

N - The number of CIDS/ blockchain nodes in the framework

$Node_i$ - Node i, where there are N nodes in framework

$t_{0,i}$ - Time when $Node_i$ joined the framework

$t_{curr,i}$ - Current time when the $Node_i$'s events are being verified

T_i - Trustscore of $Node_i$ – the trust score will be in the range – [1 - 100]

T_{init} - Initial trust score to be assigned to a new node – 50

T_{thld} - Threshold limit below which the node is not trusted anymore – 40

(PU_i, PR_i) – Public and private key pair for $Node_i$

L_s – List of suspended nodes from framework

Traffic log - Refers to the details related to an event analyzed by CIDS

$Event_{output}$ - Refers to the classification (alert or benign or) output of an event by CIDS node

Trust score increase:

$$T_i = T_i + \frac{\sum_{t_{0,i}}^{t_{curr,i}} valid\ alerts}{\sum_{t_{0,i}}^{t_{curr,i}} Events} \quad (1)$$

Trust score decrease:

$$T_i = T_i - \frac{\sum_{t_{0,i}}^{t_{curr,i}} invalid\ events}{\sum_{t_{0,i}}^{t_{curr,i}} Events} \quad (2)$$

Step 1: Initialization of a new node in the framework:

Input: New Node to be added to the framework

1. If the node is in L_s
2. do not add node to the framework
3. Else
4. Add the node as a node in the framework – assign (PU_i, PR_i)
5. Initialize the trust score of the node, T_i to 50
6. Broadcast the new node's identity (PU_i) and trust score (T_i) to the blockchain network

Output: Verified Node's addition to the framework

Step 2: Identifying the compromised nodes:

The consensus is used to verify the validity of classifications of the events. Based on the classification by the CIDS node, two cases arise during verification, false positives, and false negatives. The approach discusses the following two ways to address these cases.

Step 2.1: Detecting false positives

Input: Traffic logs to CIDS nodes

1. For every alert detected at every node:
2. Broadcast (Message = (Alert || Traffic log || T_n)PR_j)
3. Consensus mechanism at every node:
4. If (checkValidity(Alert, Traffic Log) == Valid)
5. Notify nodes for the Alert, raise an alarm
6. Trust Score increase ($Node_i$)
7. Add alert in the list of approved transactions
8. Else
9. Trust score decrease ($Node_i$)
10. If ($T_n < T_{thld}$)
11. Add $Node_i$ to L_s
12. Add a new node to replace the Suspended node

Output: Trust score increase to an honest node, identification of malicious node and penalty

Step 2.2: Detecting False Negatives

Input: Requested traffic logs from a CIDS node

1. For every $Node_j$ in CIDS:
2. Generate a $Node_i$ with $P(x = i) = \frac{1}{n} \; \forall \; x \in [1, n]$
3. For the $Node_i$:
4. Request event classification for between $t_{0,\,i}$ to time with $P(x = t) = \frac{1}{t} \; \forall \; t \in (t_{0,i}\,, t_{curr,i}]$
5. For every event in requested time period for $Node_i$:
6. If (checkValidity(Alert, Traffic Log) == Invalid)
7. Broadcast (Message = (Alert || Traffic log || T_n)PR_j)
8. Consensus (Add alert in the list of approved transactions)
9. Notify nodes for the Alert, raise an alarm
10. Trust Score decrease ($Node_i$)
11. If ($T_n < T_{thld}$)
12. Add $Node_i$ to L_s
13. Add a new node to replace the Suspended node

Output: Trust score increase to an honest node, identification of malicious node and penalty

Check Validity (Event, Traffic Log):

1. $Event_{output}$ = IDS (Traffic Log)
2. If ($Event_{output}$ == Event)
3. return “Valid” event classification
4. Else
5. return “Invalid” event classification

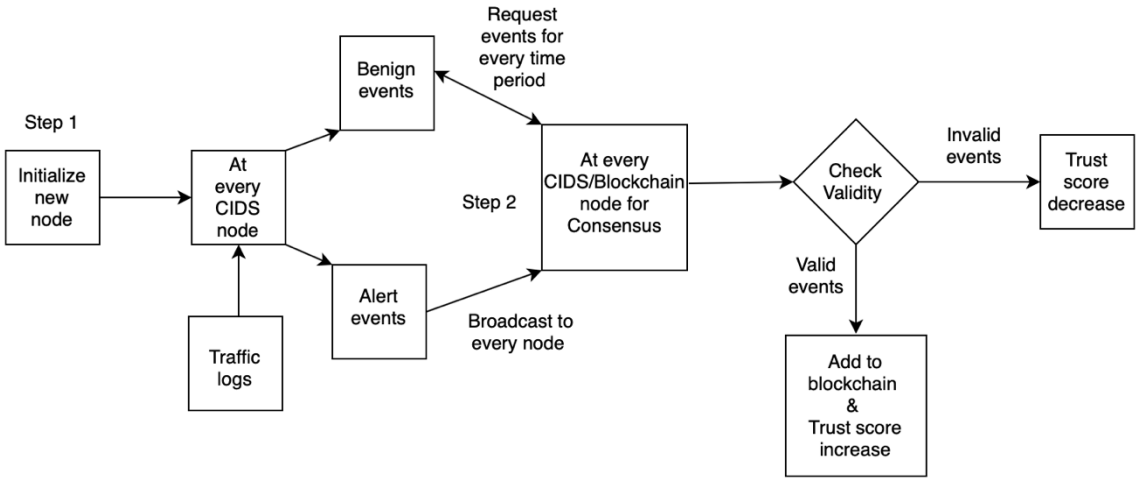

Figure 5: Overall Approach

CHAPTER 5

VALIDATION OF NETWORK TRAFFIC CLASSIFICATIONS

The Check Validity function is the key function used in our approach, which is an essential function in the consensus model. Every node's IDS component uses the check validity functionality to verify the classifications of the alerts by other nodes in the framework. All the valid alerts verified by this function are included in the list of approved transactions in blockchain. The check validity is a crucial decision-making step in identifying the compromised nodes by increasing or decreasing the trust scores based on the verified results of classifications by the nodes. This chapter focuses on explaining the approach using an example for identifying compromised nodes in the framework. Consider the scenario where a malicious node N_m is joining the framework. The following explanation using an example shows the steps in approach that will identify the node.

Step 1: Initialization of a new node in the framework

New nodes joining the framework are required to have the identity verified before being added to blockchain. If the malicious node N_m, was previously suspended is attempting to join the network, the node's identity is checked (for example, checking the node's MAC address). If it is present in the list of suspended nodes, it will not be authorized to join the framework. However, if the node was is not in the list of suspended nodes, it will be authorized to join the framework, and will be initialized as a CIDS node and a blockchain node in the framework by assigning a public and private key pair which the node will use to digitally sign the transactions in the blockchain. This same public-private

key pair can be used to identify the classifications of this node N_m and will help in identifying the malicious node in the second step. This new node's addition to the framework is notified to all the nodes in the framework so that the nodes in the framework are aware of the node's trust score.

Step 2: Consensus

Trust score calculations: The trust scores of nodes are used to assess the reliability of the node and to verify the credibility of the node's classification of the events. The trust score of a node is increased with every valid report of an event classified as an alert. The validation of an event takes place through the consensus on blockchain, where every node will verify the event through the CIDS component. The trust score of a node is decreased in the following two cases:

Case 1: Classified an event as a benign event and did not report it as an alert (false negative)

Case 2: Generated a benign event and reported it as an alert (false alarm)

The increase of trust score is calculated based on equation 1 in chapter 4 and trust score is decreased based on equation 2 in chapter 4.

In the equations, an invalid event refers to classification of the events discussed in case 1 and case 2 above. The total events are the total aggregated events monitored and analyzed by the CIDS node until the point of trust score calculation.

The Check Validity function is the function in the consensus model on the private blockchain. This function is implemented by every node as part of the consensus, for verifying the classification of the events by every node. The inputs for this function are the event classification reported by the node and traffic logs corresponding to that event. In step 1 of this function, the input traffic log is analyzed by the IDS component of every node. This generates an $Event_{output}$ which is compared against the input event from the other nodes. If both the classification matches, the function returns the classification as valid, else invalid classification is returned.

Step 2.1: Detecting false positives

The input to this step is the traffic logs to be analyzed by a CIDS node. When a CIDS node detects an event, which it considers as a valid alert, the alert is broadcasted to every node to be included in the list of approved transactions in blockchain. However, for the alert to be included in the list of approved transactions, every node in the blockchain should verify the validity of the event before moving to the next steps in the consensus. If the malicious node N_m generates false positives, the check validity function in step 4 is used to verify the validity of the event. The node N_m is required to send traffic log supporting the event's classification. If the alert is verified as valid by the check validity function, and a consensus is achieved by every node in the framework on the verification, an alarm is raised to notify about the alert. Any false positives will be detected using this check validity function in step 4, and the trust score is decreased in step 9, based on equation 2. Further, in step 10, after the trust score decrease of the node N_m, if the node's

threshold is lesser the trust score threshold, then the node is identified as compromised and suspended from the framework and added to the list of suspended nodes for future reference and verification.

Step 2.2: Detecting false negatives

False negatives occur when a CIDS node does not report valid alerts and classify the alerts as benign events. To verify this, every node ($Node_j$) in the blockchain, selects a random peer node $Node_i$ (where the probability of each node being picked is equal) requests traffic logs between a random time period. If Node N_m, was not identified in step 2.1, it will be identified in step 2.2 if the node is generating false negatives. In step 6, the check validity function in used to verify the classification of these traffic logs. If any event from these traffic logs which was a valid alert was not reported by the CIDS node N_m, the $Node_j$ will raise an alarm to alert other CIDS nodes. In step 7, the $Node_j$ will also broadcast the alert to other nodes to receive consensus to add the alert to the list of approved transactions in blockchain. In step 8, as a part of the consensus, every node will verify the alert using the check validity function. Further, in step 10, the trust score of the $Node_m$ is decreased based on equation 2. If the trust score of the node N_m is below the trust score threshold, the node is identified as compromised and suspended and added to the list of suspended nodes for future reference and verification.

Additionally, the key features of blockchain help in addressing the requirements of CIDS. The following explains how the features of blockchain and our approach presented using blockchain address the requirements of CIDS:

Accuracy: By implementing the trust-based consensus approach on the blockchain and quickly identifying and suspending the nodes causing false negatives and false positives, the overall accuracy of the CIDS is increased.

Accountability: By using a private blockchain, every node's identity is verified before adding the node to the blockchain. Additionally, the alert events classified by every node are stored in the blockchain, holding the nodes accountable to every transaction on blockchain

Integrity: Blockchains are immutable digital ledgers, which ensures the integrity of trust scores, validated traffic alerts stored on the blockchain.

Consensus: Blockchain's consensus models fulfill this requirement of CIDS

Overhead: The blockchain is used to store and share only validated traffic alerts, thus decreasing the overhead of storing every traffic log on the blockchain.

Privacy: Using a private blockchain ensures the confidentiality of the critical traffic alerts and private information in the traffic logs stored on blockchain.

CHAPTER 6

EVALUATION

Our approach has been simulated on a network of CIDS and blockchain nodes validate our approach. The approach has been implemented in python, where the functions have been written for trust score increase, trust score decrease, checking the validity of the classification by the CIDS nodes, the consensus among the nodes for detecting the false positives, and false negatives. In the network simulation, the traffic is generated continuously, which is processed by the nodes, and the classification is broadcasted to obtain the consensus before adding the events to the approved list of transactions in blockchain. The nodes have been assigned a probability of acting maliciously, and the approach was applied to see the effect of the consensus-based approach to detect false positives and false negatives generated by each node. The approach has been simulated with various changing constraints, trust scores of the nodes, the probability of the node being malicious, the number of nodes, and the number of iterations. The resulting trust scores of the nodes have been plotted against the number of iterations of the approach to observe the behavior of malicious nodes and how the trust scores change. The nodes' trust scores have been calculated continuously over 600 iterations, with the trust score threshold for 30 and above 40.

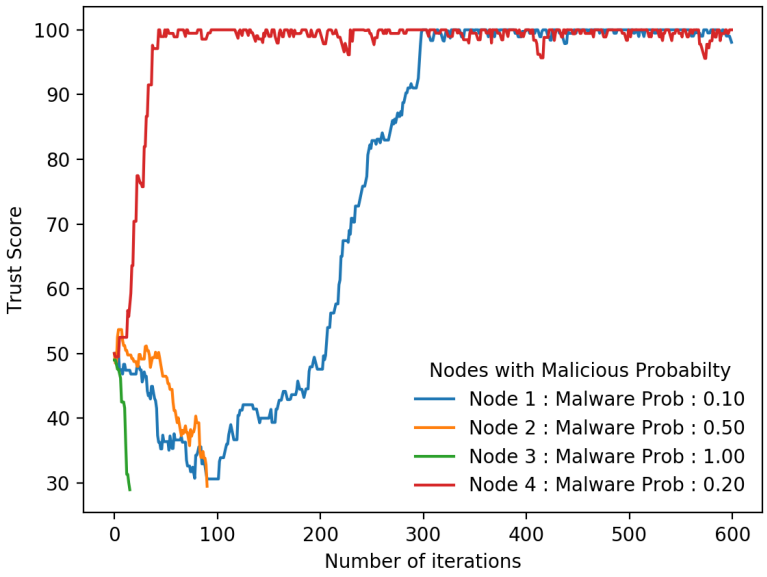

Figure 6: Approach for Four Nodes and Trust Score Threshold 30

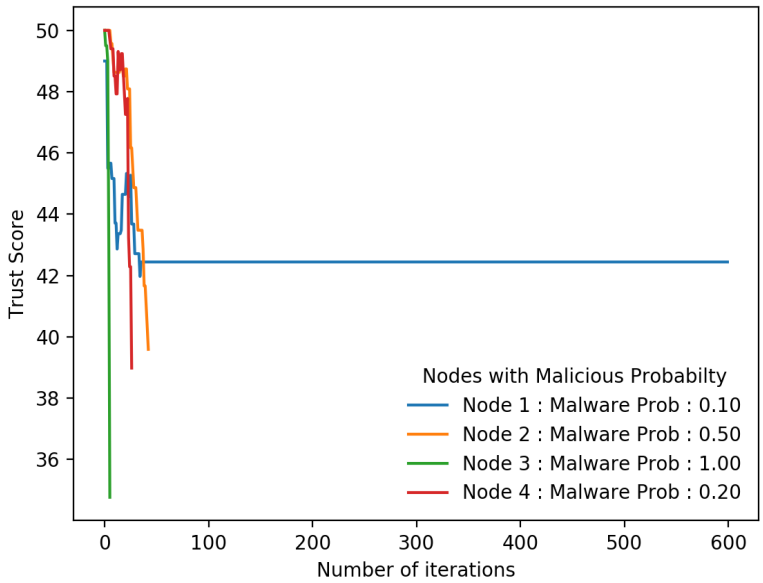

Figure 7: Approach for Four Nodes and Trust Score Threshold 40

Figure 6 and figure 7 represent the approach simulated for four nodes for a trust score threshold of 30 and 40. The probability of nodes being malicious is changed to see how this affects the trust score. The probabilities of the four nodes being malicious are 0.1, 0.5, 1, 0.2. It can be observed from the figures that the nodes with the probability of 0.5 and 1.0 to be malicious, do not have increasing trust scores in figure 6. The trust score drops sharply from 50, and once the trust score falls below 30, the nodes are suspended and no longer appear in the graph. However, using a trust score threshold of 40, even the honest nodes are affected, as the nodes with the probability of 0.5 and 1.0 to be malicious are quickly suspended, and are removed from the consensus as well. Based on the analysis of these two figures, the optimal value of the trust score threshold is 30.

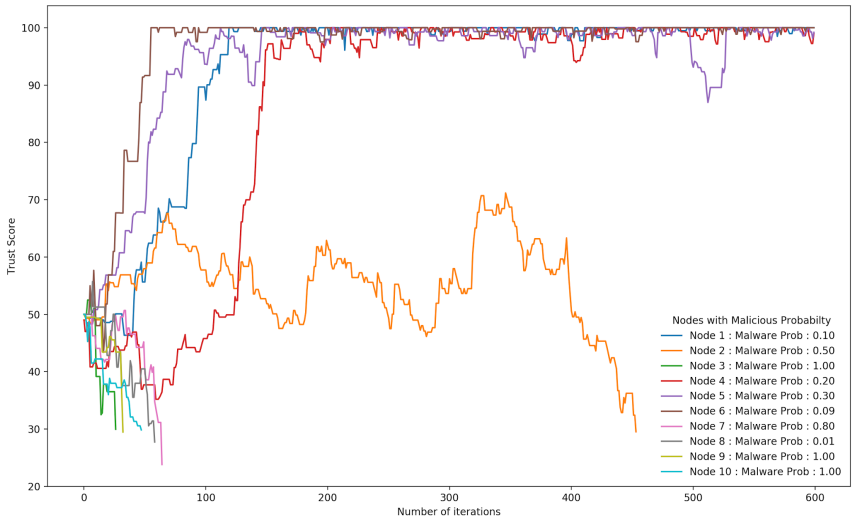

Figure 8: Approach for Ten Nodes and Trust Score Threshold 30

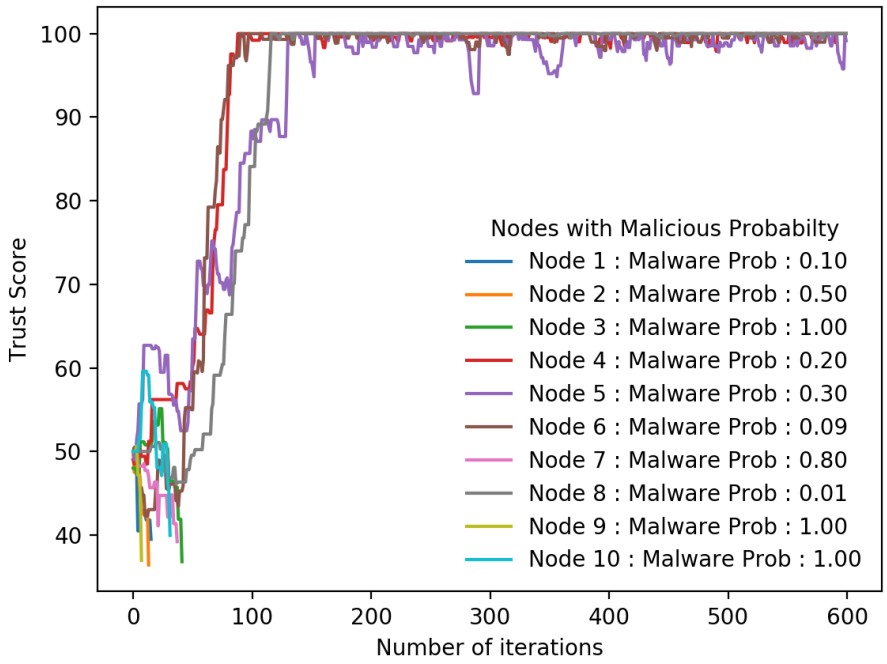

Figure 9: Approach for Ten Nodes and Trust Score Threshold 40

Figure 8 and figure 9 represent the approach simulated for ten nodes for a trust score threshold of 30 and 40. It can be observed from figure 8 that there is a gradual increase in the trust scores of the nodes when the trust score threshold is set to 30, and the decrease in the trust score is gradual as well. From figure 9, it can be observed that there is a very sharp decrease in the trust scores of the malicious nodes, as they are quickly suspended from CIDS as well as blockchain consensus.

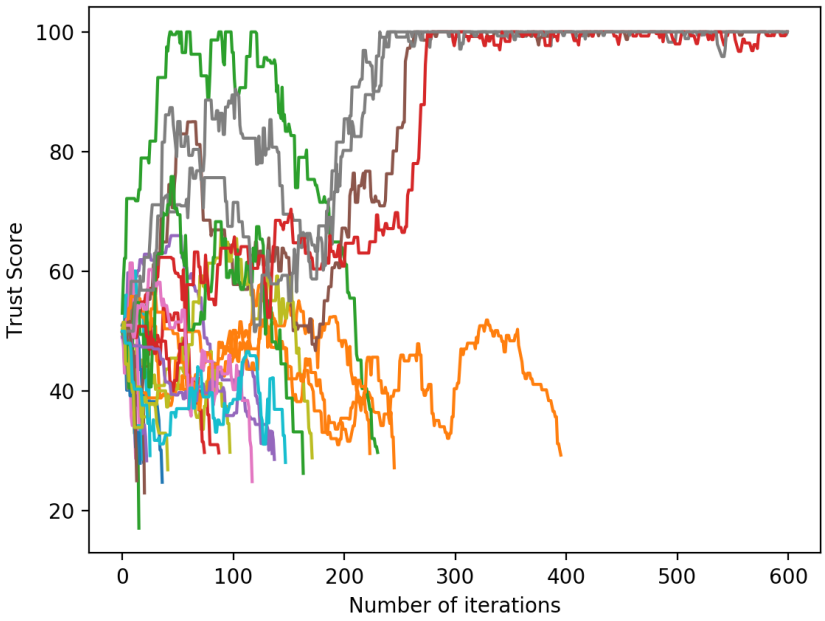

Figure 10: Approach for Thirty Nodes and Trust Score Threshold 30

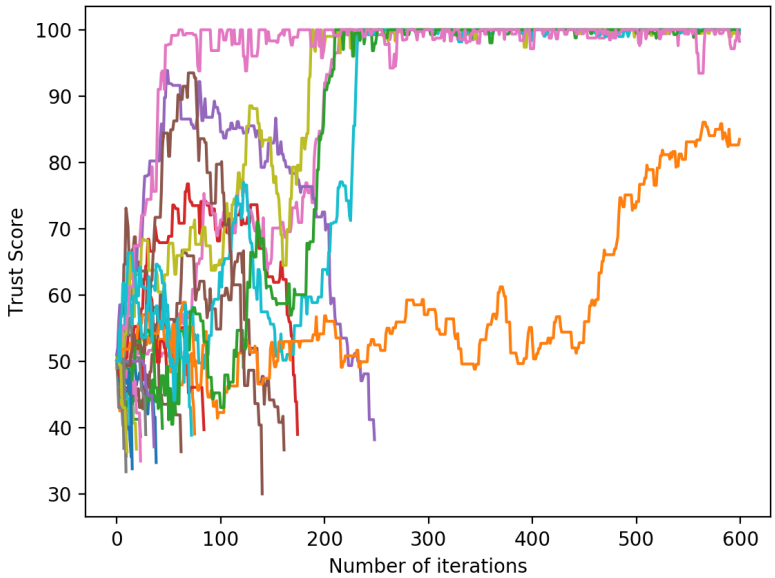

Figure 11: Approach for Thirty Nodes and Trust Score Threshold 40

Figure 10 and figure 11 represent the approach simulated for thirty nodes for a trust score threshold of 30 and 40. The observation from figure 10 is that the nodes whose trust score dropped below 40, can still participate in the consensus and affect the accuracy while still being malicious. In figure 11, where the trust score threshold is 40, the malicious nodes are quickly dropped following the sharp decreases in the trust scores. Based on the implementation results and the observations from the visualizations, it can be evaluated that, as the number of nodes increase, a trust score threshold of 40 is optimal at a higher number of nodes.

The Approach helps in addressing the insider attacks: Newcomer attacks, sybil attacks, and betrayal attacks. The significant contribution by this approach is to identify the nodes generating false positives and false negatives and suspending them from the CIDS and increasing the overall accuracy of CIDS. The following discussion shows how the insider attacks are addressed using the approach.

Sybil attacks: The nodes are required to verify their identity before joining the private blockchain. By using a private blockchain that verifies the identities of the nodes before allowing the nodes to participate in the consensus, it discourages the attackers from creating a large number of malicious identities.

Newcomer attacks: This attack is an extension of the Sybil attack, where a large number of malicious nodes created, attempt to tip-off the trust computation model. However, in this approach, the new nodes are given only a trust value of 50, and the nodes are required to build the trust value, before showing any significant influence on the consensus model. This prevents the effectiveness of the newcomer attacks. Additionally, if the new nodes

start malicious activity before building trust, the trust scores drop is sharp, and nodes get suspended.

Betrayal attacks: Betrayal attacks occur when nodes with high trust scores begin to act maliciously. However, in both cases, when the nodes with high trust scores begin to act maliciously, their trust scores will drop quickly, in both the cases of generating false positives or false negatives. It can be observed from the figures 1 - 6, that even the nodes with high trust scores will have a drop in the trust score right when they begin to act maliciously. This forces the node to stay honest to avoid the sharp decrease of the trust scores.

From the evaluation of our approach, it can be concluded that malicious nodes are quickly identified and suspended from the framework. This increases the trust among the nodes, which in turn increases the data sharing among the nodes. By detecting and addressing false positives and false negatives generated by the compromised nodes, the approach increases the overall accuracy of CIDS.

CHAPTER 7

CONCLUSION AND FUTURE WORK

From the simulation and the evaluation results, it can be observed that the approach quickly identifies malicious nodes generating false positives and false negatives. The trust scores of the malicious nodes decrease sharply. By using this approach, the insider attacks in CIDS, newcomer attacks, sybil attacks, and betrayal attacks are addressed. By identifying the malicious nodes and quickly and suspending them from the CIDS, the approach helps in increasing the overall accuracy of the CIDS.

While there are advantages of using this approach implemented on blockchain the shortcoming of the approach are as follows:

Storing the alerts on the blockchain requires every node in the framework to have a local copy of the blockchain. This gives rise to an increased overhead for storage. The approach does not yet address the internal attack which rarely occurs, collusion attack, [5] where the malicious nodes collaborate by communicating over a covert private channel.

The future works of this approach are the following:

1. Reduce the overhead of storing alerts in blockchain. This can be done by using techniques such as off-the-chain storage of the alerts, bloom filters, [28] etc.

2. Expanding the approach to consider the nodes distributed geographically and connected for collaboration only through blockchain.

By extending the scope of the approach to address the challenges mentioned above, this approach will optimize and greatly increase the performance, accuracy, reliability, and collaborative capabilities of CIDS.

REFERENCES

[1] Leuth, Knud Lasse. “State of IoT 2018: Number of IoT devices now at 7B – accelerating”. iot-analytics.com. https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/ (accessed April 5, 2020)

[2] Capital One. “Information on. Capital One Cyber Incident”. capitalone.com. https://www.capitalone.com/facts2019/ (accessed April 5, 2020)

[3] Sabahi, Farzad, and Ali Movaghar. "Intrusion detection: A survey." In *2008 Third International Conference on Systems and Networks Communications*, pp. 23-26. IEEE, 2008.

[4] Vasilomanolakis, Emmanouil, Shankar Karuppayah, Max Mühlhäuser, and Mathias Fischer. "Taxonomy and survey of collaborative intrusion detection." *ACM Computing Surveys (CSUR)* 47, no. 4 (2015): 1-33.

[5] Fung, Carol J. "Collaborative Intrusion Detection Networks and Insider Attacks." *JoWUA* 2, no. 1 (2011): 63-74.

[6] Zheng, Zibin, Shaoan Xie, Hongning Dai, Xiangping Chen, and Huaimin Wang. "An overview of blockchain technology: Architecture, consensus, and future trends." In *2017 IEEE international congress on big data (BigData congress)*, pp. 557-564. IEEE, 2017.

[7] Asif, Muhammad K., Talha A. Khan, Talha A. Taj, Umar Naeem, and Sufyan Yakoob. "Network intrusion detection and its strategic importance." In *2013 IEEE Business Engineering and Industrial Applications Colloquium (BEIAC)*, pp. 140-144. IEEE, 2013.

[8] Rowe, Neil C., and Han C. Goh. "Thwarting cyber-attack reconnaissance with inconsistency and deception." In *2007 IEEE SMC Information Assurance and Security Workshop*, pp. 151-158. IEEE, 2007.

[9] Al-Saleh, Mohammed I., Ziad A. Al-Sharif, and Luay Alawneh. "Network Reconnaissance Investigation: A Memory Forensics Approach." In *2019 10th International Conference on Information and Communication Systems (ICICS)*, pp. 36-40. IEEE, 2019.

[10] Garcia-Teodoro, Pedro, Jesus Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. "Anomaly-based network intrusion detection: Techniques, systems and challenges." *computers & security* 28, no. 1-2 (2009): 18-28.

[11] Aydın, M. Ali, A. Halim Zaim, and K. Gökhan Ceylan. "A hybrid intrusion detection system design for computer network security." *Computers & Electrical Engineering* 35, no. 3 (2009): 517-526.

[12] Bye, Rainer, Seyit Ahmet Camtepe, and Sahin Albayrak. "Collaborative Intrusion Detection Framework: Characteristics, Adversarial Opportunities and Countermeasures." In *CollSec*. 2010.

[13] Nakamoto, Satoshi. *Bitcoin: A peer-to-peer electronic cash system*. Manubot, 2019.

[14] Yaga, Dylan, Peter Mell, Nik Roby, and Karen Scarfone. "Blockchain technology overview." *arXiv preprint arXiv:1906.11078* (2019).

[15] Dhumwad, Saurabh, Mandar Sukhadeve, Chetan Naik, K. N. Manjunath, and Srikanth Prabhu. "A peer to peer money transfer using SHA256 and Merkle tree." In *2017 23RD Annual International Conference in Advanced Computing and Communications (ADCOM)*, pp. 40-43. IEEE, 2017.

[16] Zheng, Zibin, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. "Blockchain challenges and opportunities: A survey." *International Journal of Web and Grid Services* 14, no. 4 (2018): 352-375.

[17] Salimitari, Mehrdad, and Mainak Chatterjee. "A survey on consensus protocols in blockchain for iot networks." *arXiv preprint arXiv:1809.05613* (2018).

[18] Ye, Congcong, Guoqiang Li, Hongming Cai, Yonggen Gu, and Akira Fukuda. "Analysis of security in blockchain: Case study in 51%-attack detecting." In *2018 5th International Conference on Dependable Systems and Their Applications (DSA)*, pp. 15-24. IEEE, 2018.

[19] Meng, Weizhi, Elmar Wolfgang Tischhauser, Qingju Wang, Yu Wang, and Jinguang Han. "When intrusion detection meets blockchain technology: a review." *Ieee Access* 6 (2018): 10179-10188.

[20] Barvin, P. Ayesha, and G. Bakkiyaraj. "Improving Trust Management for Effective Collaborative Intrusion Detection Network." (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (2), 2015, 1500-1504

[21] Duma, Claudiu, Martin Karresand, Nahid Shahmehri, and Germano Caronni. "A trust-aware, p2p-based overlay for intrusion detection." In *17th International Workshop on Database and Expert Systems Applications (DEXA'06)*, pp. 692-697. IEEE, 2006.

[22] Fung, Carol J., Jie Zhang, Issam Aib, and Raouf Boutaba. "Dirichlet-based trust management for effective collaborative intrusion detection networks." *IEEE Transactions on Network and Service Management* 8, no. 2 (2011): 79-91.

[23] Alexopoulos, Nikolaos, Emmanouil Vasilomanolakis, Natália Réka Ivánkó, and Max Mühlhäuser. "Towards blockchain-based collaborative intrusion detection systems."

In *International Conference on Critical Information Infrastructures Security*, pp. 107-118. Springer, Cham, 2017.

[24] Vasilomanolakis, Emmanouil, Michael Stahn, Carlos Garcia Cordero, and Max Mühlhäuser. "Probe-response attacks on collaborative intrusion detection systems: Effectiveness and countermeasures." In *2015 IEEE Conference on Communications and Network Security (CNS)*, pp. 699-700. IEEE, 2015.

[25] Shala, Besfort, Ulrich Trick, Armin Lehmann, Bogdan Ghita, and Stavros Shiaeles. "Novel trust consensus protocol and blockchain-based trust evaluation system for m2m application services." *Internet of Things* 7 (2019): 100058.

[26] Wang, S., Yuan, Y., Wang, X., Li, J., Qin, R. and Wang, F.Y., 2018, June. An overview of smart contract: architecture, applications, and future trends. In *2018 IEEE Intelligent Vehicles Symposium (IV)* (pp. 108-113). IEEE.

[27] Sousa, Joao, Alysson Bessani, and Marko Vukolic. "A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform." In *2018 48th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, pp. 51-58. IEEE, 2018.

[28] Artan, N.S., Sinkar, K., Patel, J. and Chao, H.J., 2007, November. Aggregated bloom filters for intrusion detection and prevention hardware. In *IEEE GLOBECOM 2007-IEEE Global Telecommunications Conference* (pp. 349-354). IEEE.

APPENDIX A

SOURCE CODE FOR IMPLEMENTATION

The simulation code is available for public access. The entire source is available in a public Github repository through the following link:

https://github.com/Chandralekha393/Blockchain-based-approach

BIOGRAPHICAL SKETCH

Chandralekha Yenugunti is M.S student in Computer Science with concentration in Cybersecurity at Arizona State University. She received her bachelor's degree in Computer Science and Engineering from National Institute of Technology, Calicut, India. She is currently a teaching assistant for Software Security course and was also teaching assistant to courses Information Assurance and Security and Principles of Programming Languages. She was also research assistant from May, 2019 to December, 2019 under Professor Stephen S. Yau in Center for Cybersecurity and Digital Forensics, ASU. Her research interests include intrusion detection systems, network security, blockchain and cybersecurity. She has accepted an offer for full-time position of Security Analyst with Oracle America, Inc. in Global Product Security group at Oracle HQ, California.