Blockchain With AI Support: A Consensus Protocol That Recognizes Outliers for IoT Networks Based on Blockchain

Roopa S Kumar
Assistant Professor, CSE Department
Sarabhai Institute of science and Technology
doi.org/10.64643/IJIRTV12I10-204456-459

Abstract*—A new framework using machine learning for secure consensus in IoT networks based on blockchain. Despite its suitability for IoT applications, Hyperledger Fabric struggles with malicious activities in untrustworthy environments. The solution is an AI-enabled blockchain (AIBC) that uses an outlier detection algorithm within a two-step consensus protocol. In the first stage, a supervised machine learning method is used to identify anomalous activity. The Practical Byzantine Fault Tolerance (PBFT) protocol is then used for ledger updates. The performance results show that the AIBC network improves the fault tolerance of Hyperledger Fabric, with only a slight impact on the delay performance.

I. INTRODUCTION

The extensive use of IoT devices is allowing for the automation of different elements in our everyday lives. A notable achievement of IoT is the automation of residences and urban areas, known as smart homes and smart cities. A major challenge in achieving the full potential of smart homes is protecting the transmitted and saved information within the home network from harmful actions that aim to disrupt someone's residence. While numerous competing technologies attempt to shield data in smart homes from threats, blockchain has emerged as possibly the most effective for both i) guarding the home network against attacks that manipulate stored information and ii) offering a secure environment for all devices within the network to communicate with one another. In a blockchain, the data remain unchangeable owing to the foundational consensus protocols, which validate each transaction by all nodes. Consequently, manipulation attacks on the data sent or stored are not feasible through just one compromised node; a majority of nodes need to be breached for an attack to succeed. Various consensus protocols and how they apply to IoT networks can be discovered in Employing blockchain technology for Internet of Things (IoT) devices within a smart home setting is quite complex. This complexity arises mainly because IoT devices often have limited resources and may struggle to execute the heavy computations needed for consensus. For instance, consensus mechanisms like Proof of Work rely on solving demanding hash functions, require storage capacity, and depend on quick communication links. However, devices with limited resources are unable to meet these demands. Recently, various innovative consensus methods have been introduced to address these challenges. One initiative that aims to tackle these issues is the Hyperledger project, which we selected as our testing platform for this study.

A. Three-layer Structure

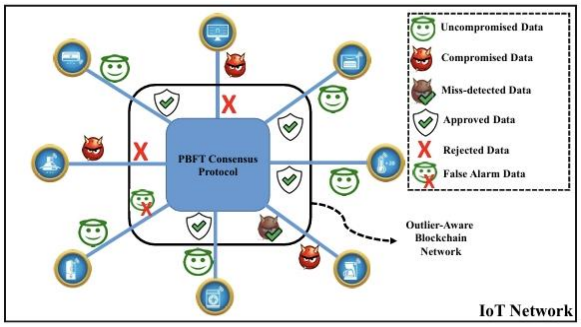


Fig.1: The two-step consensus method in our suggested AIBC network.

In this representation, the attack ratio is 37.5%, which PBFT cannot handle in a traditional blockchain setup. Nevertheless, the AIBC network that includes an outlier detector will reduce the attack's impact to 20%

in the initial phase, allowing it to effectively pass through PBFT as the secondary consensus protocol.

This paper has pointed out the potential for applying machine learning methods to the consensus process in blockchain as a direction for future research without conducting any examinations. This paper has explored the advantages of merging blockchain with artificial intelligence (AI) and proposed that a blockchain managed by a machine learning algorithm could identify attacks and implement appropriate defense measures or isolate compromised elements. We have effectively articulated and applied this concept, known as AI-enabled blockchain (AIBC).

This paper has introduced a utility function similar to that in to identify anomalies. This paper asserts that this value can serve as input for a supervised machine learning algorithm to evaluate the likelihood of an attack and hinder the confirmation of that transaction in the consensus procedure. However, he did not provide an algorithm or plan for creating a practical consensus protocol.

To test the effectiveness of our two-step consensus method for IoT networks, we construct a three-layer framework utilizing Hyperledger fabric. The first layer consists of the application layer, which contains various IoT devices. The second and third layers make up the edge blockchain layer and the core blockchain layer, which include the different elements of AIBC.

The paper assesses the latency in different segments of our implementation and quantify the number of outlier devices within each time unit using synthetic data arranged in a matrix. We contrast our proposed architecture with the standard implementation of Hyperledger fabric and examine how the outlier detection algorithm influence's fault tolerance. Findings indicate that our suggested AIBC network can achieve consensus on new data in milliseconds and offers better fault tolerance compared to a basic Hyperledger fabric implementation. This is accomplished by identifying the malicious devices in the first consensus phase and preventing them from engaging in the PBFT stage.

The structure of the 3-layer network we implemented is depicted in Fig. 2. The initial layer is the application layer, which comprises n smart devices, smart meters, and sensors (A1 to An) in a smart home network. The next layer is the edge blockchain layer, consisting of m endorsing peers (Pe1 to Pem) and data aggregators. This particular layer is tasked with endorsing new transactions (Tx1 to Txn) from the applications and plays a partial role in the 2-step consensus process. The final layer is the core blockchain layer, which includes an ordered and z regular peers (P1 to Pz). This layer is responsible for forming a block from the endorsed transactions (R1/E1 to Rn/En, where R and E signify transaction outcomes and their associated endorsements, respectively) received from the application layer, and it also contributes to the 2-step consensus process.

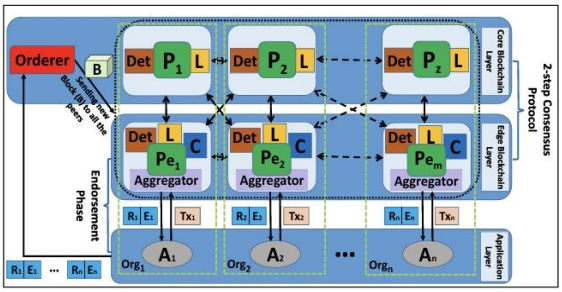


Fig. 2: The topology of our implemented 3-layer AIBC network.

In the AIBC network, there are n organizations (Org1 to Orgn), each containing one application, at least one endorsing peer (with n being less than or equal to m), and either no regular peers or just a few (with n being less than or equal to z or n being more than or equal to z). For simplicity in illustration, only one regular and one endorsing peer are shown for each organization in Fig. 2. Endorsing peers possess an aggregator to collect data (transactions) from the application layer, a version of the chaincode (called a smart contract, denoted as C) for endorsing new transactions, a copy of the ledger (labelled as L), and the detector (marked as Det) to participate in the 2-step consensus process. Regular peers only have one copy of the ledger and detector to engage in the 2-step consensus protocol, enhancing the overall security of the blockchain network. Both endorsing and regular peers engage in the 2-step consensus protocol. Therefore, as the number of peers (m and z) increases, a greater number of them should be involved.

B. Interaction Method in the AIBC Network

There are two distinct interaction methods in the AIBC network: the query process and the invoke process.

[1] Query Process:

In query process, an application links to any endorsing or standard peer to obtain the latest information about the ledger's current status. This process consists of

three steps: (i) the application establishes a direct connection to a peer (without an aggregator), (ii) the application submits a query request to that peer, and (iii) the peer retrieves its ledger copy and returns the results to the application.

[2] Invoke Process:
In this stage, an application establishes a link with its designated endorsing peers to initiate a modification in the ledger using its updated data. A simplified depiction of our execution featuring a single application (A1) and its paired endorsing peer (Pe1) is illustrated in Fig. 3 to clarify how the communication works during the invoke process. It is important to recognize that additional applications and endorsing peers, along with regular peers, are represented in Fig. 2. The other endorsing peers engage with their related applications and go through all the communication steps depicted in Fig. 3 (steps 1–7), while regular peers only undergo steps 5 and 6. The invoke process is divided into three stages: (i) the endorsement phase (indicated by solid lines), (ii) the two-step consensus protocol (indicated by dashed lines), and (iii) the update of the ledger (indicated by dotted lines) as shown in Fig. 3.

Endorsement Phase: This stage consists of four steps:

1. In this step, each application must connect to one or multiple endorsing peers as dictated by the endorsement policy (not randomly selected). The endorsement policy explains which endorsing peers from specific organizations must validate a transaction (Tx) proposed by an application (A). In our setup, each application interacts with only one endorsing peer, as illustrated in.2.

2. Application (A1) retrieves data from a smart device and transmits its new readings to its assigned endorsing peer (represented as Pe1) in the AIBC network.

2.1. The endorsing peer processes the new transaction (data) in its version of the chaincode to provide endorsement.

2.2. If the transaction receives a successful endorsement, the chain code temporarily modifies that endorsing peer's ledger to suggest an update.

3. The endorsing peer relays the result of the proposed transaction (R1) along with its endorsement (E1), which is secured by the endorsing peer's digital signature, back to the relevant application.

4. The application verifies all the results it receives along with their respective endorsements from the applicable endorsing peers to ensure they match with valid endorsements. Once these conditions are fulfilled, the application forwards the result with its associated endorsements to the order for inclusion.

2-step Consensus Protocol:
The consensus protocol is composed of two main steps, which involve the detector (responsible for spotting outliers) and the execution of PBFT.

5. The ordered brings together all the gathered transaction results and their corresponding endorsements into a new block and distributes it to all the peers (including both endorsing and regular) to commence the 2-step consensus process.

Step 1: Detector.
5.1(a), each peer assesses all the transactions included in the block through its detector that employs the outlier detection algorithm mentioned in Section III to identify and eliminate outlier data.

5.1(b) The detector prevents peers linked to the organization that generated the outlier data from being involved in the second consensus step. To accomplish this, the detector modifies the relay switch to inform its respective peer regarding which set of peers (both endorsing and To enter the second stage of consensus, it must connect to the regular step. This phase can limit the interference of more than 33.3% of compromised nodes within the PBFT consensus protocol to some degree; the specific percentage relies on the effectiveness of the detector (as mentioned in Section III-C).

Step 2: PBFT.
5.2 Each peer must confirm whether every transaction within the new block has received approval from all necessary peers outlined in the endorsement policy. Additionally, they must verify that the outcome of a particular transaction matches across all relevant endorsing peers. This ensures that the application remains secure and does not provide an erroneous outcome for the transaction. After the peers check the

endorsements, all transactions within the block are classified as either valid or invalid. Subsequently, the peers link to their trusted group of peers based on the relay switch obtained in the earlier step to achieve consensus on the new block using the PBFT approach.

II. PERFORMANCE EVALUATION

A traditional hyperledger fabric network can withstand up to one-third of harmful devices. In contrast, the suggested AIBC network can greatly enhance this limit through a well-crafted detector. There are two key probabilities linked to an outlier detection method: the probability of detection (Pd) and the probability of false alarm (Pfa). The probability of missing a detection, which is the opposite of the probability of detection, refers to the situation in which our method does not identify a harmful node. The probability of a false alarm relates to the case where the method incorrectly identifies a healthy node as being harmful. Based on these two metrics, the fault tolerance of our suggested framework can be elevated compared to the basic PBFT consensus method. Nonetheless, there are scenarios where the performance of our framework may fall below 33.3%, which can occur if the detector is poorly designed, leading to either a very high false alarm probability or a very low detection probability.

The effect of the designed detector on the fault tolerance of the AIBC network is shown in the following inequality:

$$F_{det} = F_{raw}(1 - P_d) + (1 - F_{raw})P_{fa} \leq 1/3.$$

In this formula, F_{det} represents the fault tolerance of the AIBC network after filtering the data with the designed detector. To ensure the successful implementation of the PBFT consensus in the second consensus step, F_{det} must be less than one-third. F_{raw} refers to the original fault tolerance of our network during the first step before conducting outlier detection, which can exceed one-third. However, in a standard Hyperledger fabric, there is no detector, and the fault tolerance (equivalent to F_{raw} in our AIBC network) should remain under one-third. The influence of the detector on the threshold was examined through practical application.

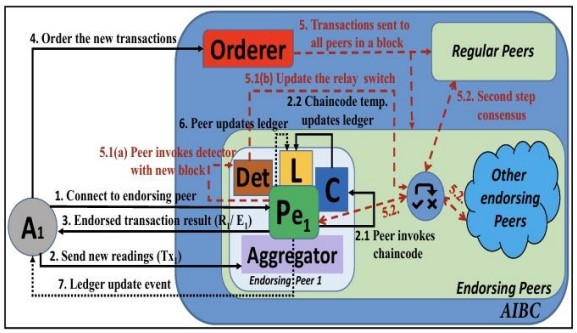


Fig. 3: The invoke process communication protocol in the implemented AIBC network using a relay switch.

III. IMPLEMENTATION DETAILS AND RESULTS

We executed the suggested 3-layer AIBC network utilizing the Hyperledger fabric framework, specifically version 1.1.0. The code was crafted in chaincode via Golang. Below are the specifications for the hardware of the two laptops employed: Core i7 6500U processor, 2.5 GHz CPU × 4, Ubuntu 18.04 LTS. Initially, we detail the structure of the established network and the dataset utilized. Subsequently, we showcase the operational performance of the AIBC network with respect to latency and the efficacy of the outlier detection algorithm.

A. Network architecture and dataset

We simulated the application layer comprising 100 sensors and devices through MATLAB. These devices transmit their data to the AIBC network at each time interval. The edge blockchain and core blockchain layers were simulated on two separate yet comparable laptops. Various elements of the AIBC network, including all peers and the ordered, are established within distinct containers using Docker. These containers are capable of communicating with one another via a channel. The two laptops' containers are linked through Docker swarm.

The developed chaincode encompasses three primary functions: init, invoke, and query. The init function facilitates the setup of the number of sensors in layer 1 along with their names, and it initializes the blockchain with the devices' initial input data. The invoke function serves to accept new data from devices in layer 1. Every time an application transmits new information to an aggregator, the invoke function is triggered, which returns the result after endorsement

back to that application. The query function acts to retrieve the current value of a device within the IoT network. The detector is also designed in Golang and includes init and invoke functions. The init function sets up the peers with the count of devices and their names and establishes the main data model in AIBC. This main data model is derived from the outlier detection algorithm applied to a synthesized dataset. We utilize a 100×100 matrix where each column represents new data gathered from an IoT network consisting of 100 devices. The invoke function updates the data model using the outlier detection algorithm, identifies outlier data within a block, and removes it. After an ordered sends a block to the peers, the invoke function of the detector is executed on each peer to carry out step-1 of the consensus protocol. This function will eliminate the outlier data from that block and transmit the result to the peer to commence step-2 of the consensus protocol.

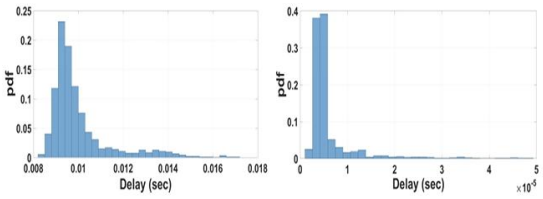
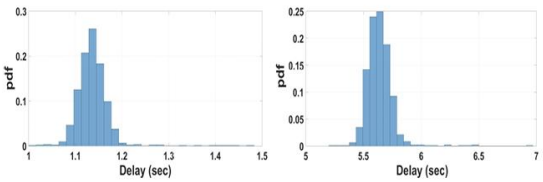


Fig. 4: Acceptable and unavoidable delays. (a) Outlier detection algorithm delay, (b) Devices state update delay.



B. Network Latency

There are various delays related to the invoke function of the designed detector: outlier detection algorithm delay, model update delay, dataset update delay, and devices state update delay. The outlier detection algorithm delay pertains to the duration required to run the algorithm to identify outlier data from the most recent 100 readings taken from 100 devices in Layer 1. Model update

Delay refers to the amount of time needed to refresh the model created by the outlier detection algorithm. It is important to note that outlier detection is a type of machine learning algorithm where the detector being learned is modified continuously based on the data gathered over time. The model gets refreshed using the most recent 100 readings collected from various devices in Layer 1. The dataset update delay is the time taken to refresh the last 100 readings from all 100 devices recorded in the ledger. The device stat update delay indicates the duration required to update new values for the devices in the ledger.

We display the probability density function (pdf) for all the delays mentioned earlier for our IoT network consisting of 100 devices over 951 time slots, as illustrated in Figs. 4 and 5. A Hyperledger fabric network reaches consensus on a new block within milliseconds, and adding that block to the ledger takes approximately 5 microseconds, as shown in Fig. 4(b). Such delays are common in any Hyperledger fabric setup and are considered acceptable for a smart home network. Our designed detector could introduce some additional delay factors for the network. The outlier detection algorithm is expected to require around 9.5 milliseconds, as illustrated in Fig. 4(a), which should not significantly impact the smart home network's functionality. However, based on Fig. 5, the model update delay and dataset update delay may take around 1.14 and 5.65 seconds, respectively, which are not suitable for a smart home network. Nonetheless, both the dataset update delay and model update delay can be easily removed by utilizing pre-learned data with a suitable dataset. Consequently, our suggested architecture results in an extra delay of approximately 9.5 milliseconds for outlier detection, which is acceptable for a smart home network since consensus is achieved in milliseconds. Fig. 6 compares the delays across different segments of our implementation.

IV. CONCLUSIONS

We suggested the AIBC network that incorporates a two-step consensus approach utilizing an outlier detection algorithm alongside PBFT. The outlier detection algorithm serves as the initial stage of consensus by confirming the consistency of incoming data and removing questionable entries to enhance the network's fault tolerance for the subsequent consensus stage (PBFT). We evaluated the latency, precision, and effectiveness of our approach. Findings indicate a notable improvement in fault tolerance of Hyperledger Fabric due to our detection mechanism.

REFERENCES

[1] T. M. Fernández-Caramés and P. Fraga-Lamas, “A review on the use of blockchain for the Internet of Things,” *IEEE Access*, vol. 6, 2018.

[2] J. Lin, Z. Shen, and C. Miao, “Using blockchain technology to build trust in sharing LoRaWAN IoT,” in *Proc. 2nd Int. Conf. Crowd Science and Engineering*, 2017, pp. 38–43.

[3] O. Novo, “Blockchain meets IoT: An architecture for scalable access management in IoT,” *IEEE Internet of Things Journal*, vol. 5, no. 2, 2018.

[4] Y. Dai, D. Xu, S. Maharjan, Z. Chen, Q. He, and Y. Zhang, “Blockchain and deep reinforcement learning empowered intelligent 5G beyond,” *IEEE Network*, vol. 33, no. 3, pp. 10–17, 2019.

[5] M. Salimitari, M. Chatterjee, M. Yuksel, and E. Pasiliao, “Profit maximization for bitcoin pool mining: A prospect theoretic approach,” in *Proc. IEEE 3rd Int. Conf. Collaboration and Internet Computing (CIC)*, 2017, pp. 267–274.