Permissioned Blockchain: Securing Industrial IoT Environments

Samira Yeasmin[1], Adeel Baig[2]
Department of Computer Engineering
Al Yamamah University, Riyadh, Saudi Arabia

***Abstract*—With the significantly increased use of the Industrial Internet of Things (IIoT), it is believed that this technology will revolutionize industrial applications and infrastructures by connecting several industrial assets. But it is getting prone to many cyberattacks and security issues. The emerging security challenges of IIoT can have a devastating effect since it deals with mission and safety-critical systems. Thus, it becomes extremely important to address the security vulnerabilities and susceptibilities of this technology. Blockchain, being one of the most significant solutions to several technologies' security problems, can play a vital role in improving the security of IIoT. Therefore, this paper proposes to use a Hyperledger Fabric Blockchain-enabled IIoT that guarantees the security of the communication medium, data storage, access, and sharing between the IIoT devices and ensures to provide limited access to the authorized identities only. This system also monitors the user access and makes sure that the transactions are performed according to their roles defined by the Certificate Authority (CA) and Membership Service Provider (MSP). Moreover, this paper presents the findings on the implementation of the blockchain network and addresses the key challenges. It evaluates the performance of the proposed network and discusses the key areas to be improved. Finally, the paper describes the benefits of the permissioned blockchain for IIoT and presents a future direction for further research and study.**

Keywords*—*Industrial Internet of Things; IIoT; permissioned blockchain; hyperledger fabric; information security; device communication; data sharing; access control

I. Introduction

Connecting numerous industrial devices to share information and make important business decisions, the Industrial Internet of Things (IIoT) is the core to aim and realize intelligent industrial manufacturing and production. It is mainly used in the mission and safety-critical systems [1] that allow making better decisions to improve the systems' efficiency. Although IIoT is believed to be competent to enhance industrial assets and digitally transform the industrial infrastructures, the centralized network creates a vulnerable environment [1]. The heterogeneous network of IIoT devices increases cyber threats including insecure IoT gateways and MQTT protocol, insecure cyber-physical systems (CPS), and SCADA [1]. Therefore, it becomes extremely important to address the security challenges and take appropriate measures to improve them.

Blockchain technology is gaining popularity in both industrial and academic research fields, showing promising results in solving the security challenges of the arising technologies. The use of blockchain in the IIoT field will improve the cyber threats and safeguard it from malicious activities that might occur during the communication between the IIoT devices. Blockchain is a Distributed Ledger Technology (DLT) that was primarily used for storing transaction information. It is a distributed network of multiple computers connected in a peer-to-peer network. The use of cryptographic security makes it suitable to secure the device communication of the IIoT network.

Taking this into account, this paper proposes to use a permissioned blockchain to secure the device communication, address and improve the security vulnerabilities of IIoT. The unique features of the permissioned blockchain including identity management and restricted user access enables only authorized parties will participate in performing transactions and device communication.

Though various studies show the implementation of Blockchain in improving IIoT security, the use of a Permissioned Blockchain is not heavily researched and implemented. The application of a Public Blockchain already exists in many forms including lightweight authentication mechanism, federated learning approach, and use of different types of encryption [2]–[9]. However, it leads to lower throughput, higher latency and resource utilization. Despite the vast literature, the implementation of Hyperledger Fabric Blockchain in securing the IIoT device communication has not been well recognized. The true benefit of a Permissioned Blockchain, the use of a Certificate Authority (CA) to issue certificates, and Membership Service Provider (MSP) to define an access control mechanism is yet to be unleashed. Validation and verification of each transaction, communication accessibility and transaction invocation to only allowed participants ensures higher throughput and lower latency, leading to an improved and secured IIoT environment.

To summarize, the main contribution of our paper is as follows: (1) this paper provides a background study on IIoT and blockchain and a comparison analysis between different open-source blockchain platforms; (2) a detailed discussion is made to state the security issues of IIoT device communication and implements the proposed idea to use a Permissioned Blockchain called Hyperledger Fabric; (3) the performance of the blockchain is analyzed and evaluated, and finally, (4) directions for future works are identified.

This paper's organization is structured as follows: Section II focuses on the background and Section III describes the need to improve the security of IIoT device communication. In

Section IV, some of the existing literature has been discussed. The proposal and a detailed discussion have been made in Sections V and VI. Section VII presents the hypothesis. The system's implementation is discussed in Section VIII. Section IX is dedicated to the evaluation of the implemented solution. Finally, Sections X and XI present the future study and conclusion.

II. Background

A. Industrial Internet of Things (IIoT)

As a subset of IoT, IIoT is defined as a connection between machines, computers, and people that enables intelligent industrial operation [10] by collecting data through wireless sensor networks, communication protocols, and internet infrastructure. This data is analyzed to produce important results that help in faster and more accurate business decisions. Although IIoT provides many benefits, it faces some challenges, especially in security, and privacy [11].

While IoT and IIoT might sound similar, Table I shows the comparison between them where the main difference is mainly in the area of interest, network, connectivity, and performance.

B. Blockchain

Blockchain is the foundation of cryptocurrency transactions and is a distributed database providing transparent, secure, and fast transactions. It is a chain of data blocks containing a time-stamp for each block [13]. It enables different parties to form and maintain consensus without an intermediary. Blockchain is decentralized, immutable, anonymous, and cryptographically sealed [14]. Table II presents the comparison between the three types of blockchain, having the main difference in network type, consensus, and read-write (RW) permissions.

C. Blockchain Platforms

Many blockchain platforms allow building decentralized applications including the open-source blockchain platforms presented in Table III. The main difference is in the type of blockchain network that also defines the transaction visibility such as public, permissioned, or private. Only Hyperledger uses a pluggable consensus protocol. Moreover, a higher throughput generates a lower latency, increasing energy and computational costs.

D. Hyperledger Fabric

Hyperledger Fabric is a permissioned blockchain where all the participants have a registered id, and all transactions are private and confidential [28], also are authenticated, authorized. It implements a distributed ledger platform to run Chaincode [28], delivering a high degree of resiliency, flexibility, confidentiality. It supports a pluggable consensus protocol [29].

1) Key Components of Hyperledger Fabric

a) Certificate Authority (CA): An CA is responsible for creating, managing, and issuing certificates to different network actors by providing them with a pair of public and private keys, restricting user access [30]. The certificates are digitally signed and bind together with the actor's public key. It issues a root and enrollment certificate and allocates a transaction certificate to each authorized member [31].

b) Membership Service Provider (MSP): It provides membership permission based on the certificates and delivers services such as identity validation, user registration, and authentication, also assign appropriate permission. It decides if the user will be a peer, admin, client, orderer, or member [30]. This component is installed on each channel peer to ensure transactions are authenticated [28]. After the CA provides a key pair and the transactions are signed using a public key, MSP verifies the transaction [30].

c) Peers [30]: The peer nodes host ledger and smart contracts, encapsulating shared processes and information.

- Endorser/Endorsing Peer: This peer validates the transaction and executes the Chaincode without updating the ledger. In the end, the endorser might approve or reject the transaction.
- Orderer Peer: It does transaction ordering, creating, and delivering new block to all the peers, eliminating bottlenecks.
- Anchor Peer: When a configuration block has updates, this peer broadcasts the updates to the rest of the peers. Anchor peers are discoverable and can be communicated by all the other peers of the network.

TABLE I. Comparison between IoT and IIoT

Area	**IoT**	**IIoT**
Focus	Consumer-level devices	Mission or safety-critical systems
Service [11]	Human-centered	Machine-centered
Architecture	3 or 5 layers [12]	3 layers [10]
Communication	Business-to-Consumer	Business-to-Business
Used in [11]	New devices & standards	Existing devices & standards
Connectivity [11]	Ad hoc	Structured
Volume of data [11]	Medium to high	High to very high
Scalability	Used in low scale network	Used in large scale network
Latency & Speed	Utility centric	High speed & minimum latency is required

TABLE II. Comparison of Three Types of Blockchain

	Public	**Consortium**	**Private**
Network	Decentralized	Partially centralized	Centralized
Consensus	Permissionless	Permissioned	Permissioned
RW	Public	Public/Permissioned	Permissioned
Example	Bitcoin, Ethereum, Litecoin	Quorum, Hyperledger, Corda	Bankchain

TABLE III. COMPARISON BETWEEN DIFFERENT OPEN-SOURCE BLOCKCHAIN PLATFORMS

Ethereum	
Blockchain Type & Network	Public/Private & Decentralized [14]
Consensus Algorithm	PoW
Cryptocurrency	Ether
Smart Contract	Written in Solidity
Vulnerability to attacks	51% attack [15]
Data Confidentiality	No
User Authentication	Digital Signature
Throughput & Latency	6-7 TPS [16] & 15-20 sec
Energy & Computational Cost	High [16]
Hyperledger	
Blockchain Type & Network	Consortium/Partially centralized [14]
Consensus Algorithm	No consensus or Pluggable consensus or Practical Byzantine Fault Tolerance
Cryptocurrency	No native cryptocurrency
Smart Contract	Written in Go, Java, Node.js
Vulnerability to attacks [17]	>1/3 faulty nodes [15] & DoS attack
Data Confidentiality	Yes
User Authentication	Based on enrolment certificates
Throughput & Latency	>1,000 TPS [18] & Less than Ethereum
Energy & Computational Cost	Low [15]
Corda	
Blockchain Type & Network	Consortium & Decentralized
Consensus Algorithm [19], [20]	Pluggable consensus, Validity Consensus & Uniqueness Consensus
Cryptocurrency	No native cryptocurrency
Smart Contract	Written in Kotlin and Java [20]
Vulnerability to attacks	Denial-of-state (DoSt) attack [21]
Data Confidentiality	Yes
User Authentication	Digital signatures
Throughput & Latency	600 TPS [22] & Low
Energy & Computational Cost	High [23]
Openchain	
Blockchain Type & Network	Private & Decentralized
Consensus Algorithm	Partitioned Consensus [24] and PoA
Cryptocurrency	No native cryptocurrency
Smart Contract	No [25]
Vulnerability to attacks	--------
Data Confidentiality	Yes
User Authentication	Digital signatures
Throughput & Latency	1000 TPS & Low
Energy & Computational Cost	High
IOTA	
Blockchain Type & Network	Public [15] & Partially centralized
Consensus Algorithm	Tip Selection Algorithm
Cryptocurrency	mIOTA
Smart Contract	No [21]
Vulnerability to attacks	34% attack [15]
Data Confidentiality	No
User Authentication	Digital signatures
Throughput [15] & Latency	7-12 TPS & Varies from mins to hours

Energy & Computational Cost	Low [15]
Ripple	
Blockchain Type & Network	Consortium & Decentralized/Centralized
Consensus Algorithm [26]	Ripple Consensus Algorithm (RPCA)
Cryptocurrency	Ripple (XRP)
Smart Contract	No [25]
Vulnerability to attacks	DoS & Theft attack [25]
Data Confidentiality	Yes
User Authentication	Digital signatures
Throughput & Latency	1,500 TPS [27] & Low
Energy & Computational Cost	Low [27]

III. Problem Statement

While IIoT devices can improve efficiency, it also comes with potential cybersecurity challenges. Since all the devices are connected, security becomes the prime concern while implementing it. The centralized nature of IIoT devices makes it open to different cyberattacks since compromising one single point can infect and destabilize the whole network. IIoT communication is transparent between the stakeholders and it makes the security problem worse as it becomes susceptible to different kinds of cyber-threats such as Man-in-the-Middle (MITM) and Denial-of-Service (DoS) attacks [32]. Communication between IIoT devices needs to be secured since they generate, process, and exchange a huge amount of data that is related to the mission and safety-critical infrastructures. A data breach may happen while sharing or transmitting data. Since IIoT devices are being used in mission and safety-critical systems, a key issue is to protect these valuable and sensitive data. Therefore, there is a need to address and improve data security by securing the communication between IIoT devices. Using Hyperledger Fabric Blockchain, the following questions were investigated: (a) Can Hyperledger Fabric provide better confidentiality and integrity compared to current approaches? (b) Does Hyperledger Fabric CA and MSP ensure access control and provide trust between devices? (c) Can a permissioned blockchain be integrated with IIoT to achieve higher throughput, lower latency, and better resource utilization?

IV. Literature Review

Authors in [2] proposed a lightweight authentication mechanism for industrial device communication using simple hashing functions to complete device authentication. In [3], authors studied the CLS scheme of [4] and presented its vulnerability towards public key replacement attacks to achieve data authenticity in IIoT. A PoW credit-based consensus algorithm is used for IIoT devices in [5]. The DAG-blockchain-based architecture included device authorization and proposed wireless sensors to act as light nodes having Private Key (PK) and Secret Key (SK) to sign transactions, and gateway and manager to act as full nodes having PK hardcoded in gateways.

The author in [6] proposed using a dynamic secret sharing mechanism in the IIoT data transmission technique using power blockchain. It included users to submit transactions and issue certificates (TCerts). Paper [7] proposed a multi-party data-sharing model using a permissioned blockchain where only registered participants could share or access data and run a consensus algorithm named Proof of training quality (PoQ) using a federated learning approach. In [8], the authors combined supply chain, IIoT, and blockchain to securely share data using attribute-based encryption. The mechanism included the registration of nodes and the definition of user roles according to the smart contract or signature provided by the admin. Authors in [9] proposed a blockchain-based IIoT architecture to improve processing power, security and privacy. Whitelist and blacklist mechanisms were used to restrict access and allow transactions via PoW.

V. Proposed Idea

A huge amount of data is generated and shared between the IIoT devices, including sensitive information. Therefore, a secure communication medium is required to enhance data security and privacy. The proposed idea is to use a permissioned blockchain that will only allow authorized members to access the network. It will permit only one or more nodes to work together to control and restrict access of members of the chain network. As a permissioned blockchain, Hyperledger Fabric can be used that allows communication only between the authorized members. The network's goal is to enforce a trusted device communication between multiple parties connected through IIoT. All the other members outside the network will be considered malicious to secure it from cyberattacks. Each member of the network is responsible for setting up their peers authorized by a CA. An MSP will allow permissions based on the CA and define access control rules. Endorsing Peer will execute transactions and Anchor Peer will update other peers. Also, Orderer Peer will create and deliver new blocks. It is important to mention here that the transactions in the entire blockchain network include data storage, access, sharing, and monitoring by the organizations' admins.

While communication happens between two different IIoT devices, only allowed participants will perform the transactions based on their roles defined by MSP. The communication medium will be secured enough as the peers will verify and validate each transaction before adding it to the network's ledger.

VI. Hyperledger Fabric Blockchain-enabled IIoT

The use of a Hyperledger Fabric blockchain ensures a secured environment where only permissioned organizations can perform transactions. One of its main features includes

organization members acting as participants. But before deciding which members will participate in the blockchain, a CA creates identities and MSP defines user access roles.

A. CA

The built-in Fabric CA [30] plays a vital role in securing the device communication of IIoT devices. A server and a client component make up the Fabric CA. It is used to create a new root CA that works as per the requirement of the system. The root CA creates an intermediate CA that generates certificates to the identities. The same database is shared among all the CA servers to keep track of identities and certificates. To register a new identity, the registrar will need an attribute along with a value because the new identity's affiliation and the registrar's attribute must be equal. If these conditions are met, the CA creates an identity and provides a keypair that consists of a public and private key. The public key is used to sign a transaction since the private key cannot be shared publicly. The MSP component ensures the verification of the transactions.

B. MSP

After registering a new identity on the CA, the MSP defines the role based on the certificate. The roles include a peer, admin, client, orderer, or member. Since the attribute and value are used to register a new identity on CA, it becomes easy for the MSP to decide a particular role for the participant. This allows an access control mechanism in the system. According to MSP, only 'client' identities can invoke transactions [30]. Whereas admins handle administrative tasks and peers take care of the transactions and ordering. While invoking a new transaction, the clients use their public key to sign the transaction and if it matches with the private key, it is added to the transaction. The MSP ordering service contains all the public keys of the clients. The verification process includes the MSP to check if the public key matches the public key it has. If verified, it is sent to the endorser peer for further processing.

C. Endorser Peer

After receiving the transaction invocation request, it checks the certificate detail and role of the transaction requester. The Chaincode is executed in this phase, following the endorser peer to decide the execution of the transaction. It is important to note that, only the endorser peer is having the Chaincode, therefore, it does not need to be installed on every node of the blockchain. Thus, it increases the scalability of the blockchain network. After deciding the validity of the transaction, the endorsement response, including the RW set, is sent to the client who invoked the transaction. If approved, the client sends the approved transaction to the orderer peer to process it.

D. Orderer Peer

The orderer peer, the central communication channel, adds the transaction into a block. Kafka is an ordering mechanism that helps in having a fault-tolerance solution for transaction ordering. To provide consistency across the whole network, the orderer peer orders all the transactions sequentially and prevents the double-spending attack. After adding the transaction into a block, it is forwarded to the organization's members to commit to the ledger. However, the verification policy is run here again to verify if the transaction has been endorsed according to the Chaincode endorsement policy.

E. Anchor Peer

The anchor peer updates and notifies the other peers about the inclusion of the new block to the ledger. The local ledger is updated with the newly added block. It helps in maintaining synchronization across the whole network.

Fig. 1 explains the process of adding a new identity to the Hyperledger Fabric Blockchain network and invoking a transaction.

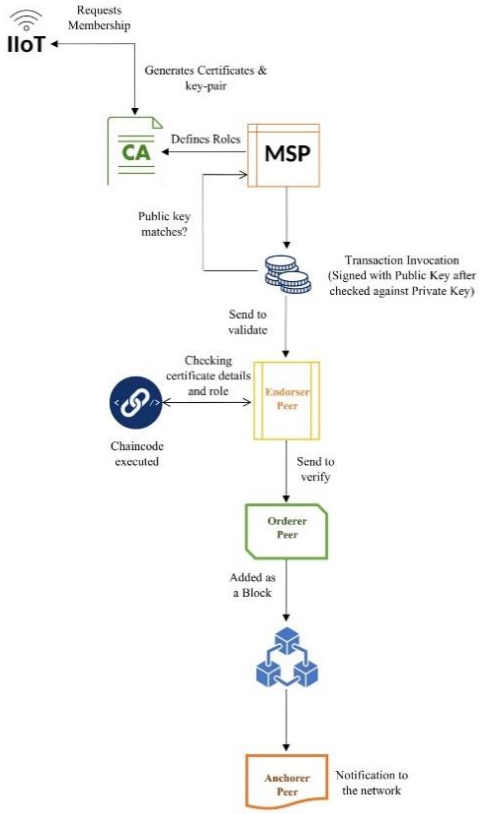

Fig. 1. Process of Hyperledger Fabric Blockchain-enabled IIoT.

VII. HYPOTHESIS

The Hyperledger Fabric Blockchain improves and enhances the IIoT network's security through the restricted participation of members of the organizations. Whenever a new organization wants to join the network, the CA generates a certificate to verify access. Moreover, the newly added organization's configuration settings and access control mechanism are defined by the MSP enabling the network to be secured from unauthorized access and transaction performance. Since the blockchain and transaction execution is restricted by the defined access control mechanism, the proposal ensures data integrity and confidentiality across the whole network. Every transaction is validated by the endorsement policy of the network, allowing only authorized parties to get involved in this process. This permissioned network is well suited for the IIoT environment to ensure data is stored, accessed, and shared only between trusted parties to achieve privacy and security.

VIII. IMPLEMENTATION

The implementation details of the proposed Hyperledger Fabric Blockchain-enabled IIoT include the addition of IIoT devices to the existing channel of the open-source project and performing transactions in a secured manner as hypothesized. In order to implement the proposed system, the open-source project Hyperledger Fabric blockchain has been chosen since it meets the requirements of our system such as CA, MSP, the three types of peer components. As Hyperledger Fabric is a permissioned blockchain and adds a level of data security and privacy, it suits the implementation requirements of this system.

As hypothesized, a CA is generated for all the Hyperledger Fabric Blockchain-enabled IIoT devices, allowing MSP to define user roles. This step allows an access control mechanism for the devices and secures the network from unauthorized access and transaction performance. Moreover, endorsers verify the transactions according to the user definition and authenticate the signatures with the CA. Therefore, CA generated certificates, the access control mechanism, and configuration updates defined by the MSP can provide a privacy protection layer across the whole network and prevent malicious network intrusion.

A. Setting up the Environment

The blockchain network has been implemented on Hyperledger Fabric v1.1.0. The experiments have been carried out on a Virtual Machine (VM). The VM acts as the IIoT environment where the Hyperledger Fabric Blockchain is configured, and IIoT devices are created. We selected VM as it gives the flexibility to implement the required configurations and contributes to our concept of a secure and limited resource IIoT environment. The hardware and software environments are described in Table IV.

TABLE IV. HARDWARE AND SOFTWARE ENVIRONMENT SPECIFICATIONS

Type	Environment	Specification
Hardware	CPU	Intel® Core™ i7-7500U CPU @ 2.70GHz
	Memory	6GB
	Hard Disk	20GB
Software	OS	Ubuntu 18.04 LTS
	docker	19.03.13
	docker-compose	1.17.1
	nodejs	8.10.0
	npm	5.3.0
	golang	1.10.4

The fabric network has been created with two organizations that are already provided by the open-source project. Both organizations consist of two endorsing peers and a CA. Each channel of the blockchain consists of a number of IIoT devices that can interact with each other through performing transactions and without any intermediaries.

B. Adding a New IIoT Device

To simulate the proposed system, three organizations act as three IIoT devices. It has been assumed that each organization plays the role of an IIoT device and each device has two running peers. Org1, Org2, and Org3 represent the devices. The Hyperledger Fabric comes with two organizations already developed in the network, Org1 and Org2. Therefore, our simulation focuses on adding a new IIoT device, Org3 that requires generating certificates and configurations.

To add a new organization, the sub-directory "first-network" of the root directory "fabric-samples" has been used. After launching and bringing up the existing IIoT network using docker-compose, crypto materials and certificates were generated for the new IIoT device. Configuration files were also prepared for crypto-config and transactions. Also, configuration materials were generated for the new device. The org3-crypto.yaml file generates keys and certificates for the new IIoT device and creates two peers. Therefore, the artifacts of the Org3, IIoT device 3, configuration file consisted of an admin user certificate, a CA, an MSP, TLS certificates, two peers, and users. The configuration files are shown in Fig. 2.

To update the processes, a configuration tool named configtxlator has been used. It performs transactions and configures tasks, providing a stateless REST API without an SDK [30]. Using a Command Line Interface (CLI) helps in encoding and decoding between protobufs and JSON. To add the new IIoT device to the existing channel of the blockchain, this tool fetches a new configuration block and updates the information inside it. This step prevents repetition in the configuration changes and adds Org3MSP to the network. This procedure updated the existing config.json file to a modified_config.json file containing the new IIoT device's configurations.

To support the concept of security and write the configuration updates on the ledger, the Org1 and Org2 admin signs on peer0.org1 and peer0.org2. Then the Orderer processes the signatures and adds a new block to the network. Therefore, the block height is changed from 5 to 6, as shown in Fig. 3. In this way, the new IIoT device gets defined in the channel and is ready to become a part of it.

To join the channel, the peers of the device need to be up and running using docker-compose. The genesis block, the first block of the network, is copied in the CLI for the IIoT device specifying the environment variables. The ordering service can verify the new device by receiving a call and successfully adds it to the channel. The new device's signature is added to the call for service while sending it to Orderer for verification purposes. Otherwise, the ordering service rejects the call. Fig. 4 represents the proposal submitted by the new device to join the existing IIoT-blockchain channel. After acceptance from Orderer, the IIoT device is added to the channel.

```
samirayeasmin@samirayeasmin-VirtualBox:~/fabric-samples/first-network/org3-arti
facts/crypto-config/peerOrganizations/org3.example.com$ ls
ca  msp  peers  tlsca  users
```

Fig. 2. IIoT Device Configuration Files.

```
root@53cedfa3aa56:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chan
nel getinfo -c mychannel
2020-10-09 09:28:00.403 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser an
d orderer connections initialized
Blockchain info: {"height":6,"currentBlockHash":"gGAkWV5ZDw9cB3p531fTG2b/dv6UIz
JPvWV4Ot5Z1V0=","previousBlockHash":"qBd/uwluvj2LrA5Ok8hZFbNFGEHgIX2CkMF1qNpZ8s
I="}
```

Fig. 3. Block Height.

```
root@ad0eb3690ab8:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chan
nel join -b mychannel.block
2020-10-10 09:12:08.171 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser an
d orderer connections initialized
2020-10-10 09:12:08.227 UTC [channelCmd] executeJoin -> INFO 002 Successfully s
ubmitted proposal to join channel
2020-10-10 09:12:08.227 UTC [main] main -> INFO 003 Exiting.....
```

Fig. 4. New IIoT Device Joining the Blockchain Channel.

C. Experimental Results

To perform some transactions, the Chaincode needs to be updated for all the peers of the devices so that Chaincode instantiation can be made. This step allows the newly added device's endorsement policy to be consistent with the rest of the devices, and it also ensures the newly added device is a valid member to endorse transaction invocation.

After specifying the endorsement policy and upgrading the Chaincode for the new IIoT device, some transactions are queried to evaluate the performance of the device communication. To begin with, a call is instantiated with a value of "a" to be 90 and "b" to be 210. The instantiation and endorsement policy is represented in Fig. 5. This allows the new device to perform transactions during the endorsement phase. Fig. 6 represents transaction execution, and it is performed between the peers of the devices. Two transactions are executed. The first one sends a value of 10 to move from "a" to "b". Therefore, the value of "a" is 80, and "b" is 220 after performing this transaction.

Likewise, Fig. 7 shows one more transaction that is invoked with a value of 30 to be moved from "a" to "b", making "a" to be 50 and "b" to be 250. This is how the devices and peers communicate with each other by performing transactions and following the endorsement policy.

```
root@bec29de5afd0:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chai
ncode upgrade -o orderer.example.com:7050 --tls $CORE_PEER_TLS_ENABLED --cafile
 $ORDERER_CA -C $CHANNEL_NAME -n mycc -v 2.0 -c '{"Args":["init","a","90","b","
210"]}' -P "OR ('Org1MSP.peer','Org2MSP.peer','Org3MSP.peer')"
2020-10-10 09:16:55.757 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001
Using default escc
2020-10-10 09:16:55.757 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002
Using default vscc
2020-10-10 09:17:21.213 UTC [main] main -> INFO 003 Exiting.....
```

Fig. 5. Endorsement Policy and Chaincode Instantiation.

```
root@bec29de5afd0:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chai
ncode upgrade -o orderer.example.com:7050 --tls $CORE_PEER_TLS_ENABLED --cafile
 $ORDERER_CA -C $CHANNEL_NAME -n mycc -v 2.0 -c '{"Args":["init","a","90","b","
210"]}' -P "OR ('Org1MSP.peer','Org2MSP.peer','Org3MSP.peer')"
2020-10-10 09:16:55.757 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001
Using default escc
2020-10-10 09:16:55.757 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002
Using default vscc
2020-10-10 09:17:21.213 UTC [main] main -> INFO 003 Exiting.....
root@ad0eb3690ab8:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chai
ncode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}'
2020-10-10 09:18:30.827 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001
Using default escc
2020-10-10 09:18:30.828 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002
Using default vscc
Query Result: 90
2020-10-10 09:18:54.976 UTC [main] main -> INFO 003 Exiting.....
root@ad0eb3690ab8:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chai
ncode invoke -o orderer.example.com:7050  --tls $CORE_PEER_TLS_ENABLED --cafile
 $ORDERER_CA -C $CHANNEL_NAME -n mycc -c '{"Args":["invoke","a","b","10"]}'
2020-10-10 09:19:24.151 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001
Using default escc
2020-10-10 09:19:24.151 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002
Using default vscc
2020-10-10 09:19:24.169 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 003 C
haincode invoke successful. result: status:200
2020-10-10 09:19:24.170 UTC [main] main -> INFO 004 Exiting.....
root@ad0eb3690ab8:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chai
ncode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}'
2020-10-10 09:19:44.609 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001
Using default escc
2020-10-10 09:19:44.609 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002
Using default vscc
Query Result: 80
2020-10-10 09:19:44.613 UTC [main] main -> INFO 003 Exiting.....
```

Fig. 6. Transaction Execution, a=80 and b=220.

```
root@ad0eb3690ab8:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chai
ncode invoke -o orderer.example.com:7050  --tls $CORE_PEER_TLS_ENABLED --cafile
 $ORDERER_CA -C $CHANNEL_NAME -n mycc -c '{"Args":["invoke","a","b","30"]}'
2020-10-10 09:19:58.633 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001
Using default escc
2020-10-10 09:19:58.637 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002
Using default vscc
2020-10-10 09:19:58.669 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 003 C
haincode invoke successful. result: status:200
2020-10-10 09:19:58.670 UTC [main] main -> INFO 004 Exiting.....
root@ad0eb3690ab8:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chai
ncode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}'
2020-10-10 09:20:10.468 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001
Using default escc
2020-10-10 09:20:10.469 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002
Using default vscc
Query Result: 50
2020-10-10 09:20:10.473 UTC [main] main -> INFO 003 Exiting.....
root@ad0eb3690ab8:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chai
ncode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","b"]}'
2020-10-10 09:38:19.344 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001
Using default escc
2020-10-10 09:38:19.344 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002
Using default vscc
Query Result: 250
2020-10-10 09:38:19.351 UTC [main] main -> INFO 003 Exiting.....
```

Fig. 7. Transaction Execution, a=50 and b=250.

IX. Performance Analysis and Evaluation

To test and evaluate the performance of the Hyperledger Blockchain-enabled IIoT, Hyperledger Caliper version 0.3.2 [33] was used. It is a benchmarking tool to evaluate any Hyperledger Fabric network's performance and provides necessary metrics to analyze the blockchain in terms of success rate, transaction throughput, transaction latency, and resource consumption, including CPU and memory usage. Hyperledger Caliper helps determine scalability, bottlenecks, anomaly, etc. issues in the developed blockchain network.

A. Performance Metrics

1) Throughput: Transaction throughput is the rate at which valid transactions are committed and executed successfully. The throughput of the blockchain network is expressed as transactions per second (TPS). The transaction throughput highly depends on the send rate of the network. The send rate is the rate at which the transactions are sent to the network for execution. It is calculated as the following formula (1).

$$Send\ Rate = \frac{(Successful\ Transactions - Failed\ Transactions)}{(Last\ Submitting\ Time - First\ Submitting\ Time)} \quad (1)$$

The Hyperledger Caliper calculates the transaction throughput of the System Under Test (SUT) using the following formula (2).

$$Throughput = \frac{Successful\ Transaction}{(Last\ Submitting\ Time - First\ Submitting\ Time)} \quad (2)$$

The last submitting time is when the transaction gets executed and the first submitting time indicates the time when the transaction was submitted for execution in the network. Throughput only represents successfully committed transactions.

It can be noticed from the above formulae that throughput highly depends on the send rate of the network. If the send rate is high, the transactions throughout will also be high since a high send rate indicates a greater number of transactions to be successfully executed across the network.

2) Latency: Latency helps to analyze the amount of time it takes for transactions to be successfully executed or failed if invalid and be effective to be usable across the blockchain network. Hyperledger Caliper calculates the number of committed transactions and when it is successfully executed or failed if invalid. The following formula (3) is used to calculate this metric for each transaction.

$$Latency = Last\ Submitting\ Time - First\ Submitting\ Time \quad (3)$$

Latency indicates the time it takes for both the successfully executed and failed transactions that were invoked. Using this formula, the Hyperledger Caliper shows the maximum, minimum, and average latency of the blockchain network.

3) Resource consumption: The Hyperledger Caliper represents the consumption of resources by each peer in terms of the total CPU used in percentage and the amount of memory used to complete their job.

B. Results Analysis

The system analysis was performed with only query transactions as they are generated to communicate with one or more peers of the network, simulating the IIoT device communication. Transaction (Tx) throughput, latency, and resource usage by the peers have been used for the performance metrics. Tables V and VI present the performance analysis results of the Hyperledger Caliper benchmarking tool.

1) Transaction throughput and latency: The transaction throughput metric in the Hyperledger Caliper demonstrates the rate of TPS and shows the number of successful transactions. Whereas the transaction latency indicates the time between the submission and execution of a transaction. As shown in Fig. 8, the proposed approach's performance analysis results indicate higher throughput with the increasing number of transactions. It is important to note that the results are taking care of the variation found while running the performance analysis multiple times. As the number of transactions increases, from 24 up to 5000, so is the throughput as Hyperledger Fabric has higher throughput [18]. Though the transactions are faster in a permissioned blockchain than a public blockchain [17], the transaction latency increases with the increasing number of transaction frequencies. If a high-performance server is used, the time required to execute each transaction will further decrease as the signing and encryption will take a shorter time. Moreover, since query transactions do not require consensus from the orderer, but arebeing handled by the peer itself using Chaincode, the throughput of the query transaction is high, and the transaction delay is comparatively lower than a Public Blockchain [17]. Transaction verification by endorser peer ensures the network's security as tempering the data of the transaction will change the hash of the block causing a smaller number of transactions to be performed and decreasing the throughput of the network. The endorsers can quickly find the tempered block to fail and invalidate. Moreover, since the network allows only a limited number of nodes in the consensus mechanism, it helps decrease the delay in response and enhances the network's performance.

Some drops and dips in the throughput and latency can be noticed in Fig. 8 graph, as the throughput highly depends on the send rate. If the send rate is high, so is the throughput. But if latency is considered, it has an inversely proportionate relationship with throughput. Therefore, the send rate indirectly affects the latency of the network while directly affecting the throughput. Thus, to produce results with greater accuracy, the send rate has been generated within a range instead of generating with a fixed rate. This allows us to simulate a real IIoT environment where communication will occur with different numbers of transactions and a non-fixed send rate helps to analyze the performance with different scenarios.

2) Resource utilization: Hyperledger caliper provides insights on the resource utilization of the blockchain network indicating the usage of CPU and memory. The Docker container of the blockchain network retrieves the container statistics and provides the results of the benchmarking. It is noticeable from Fig. 9 that our system improves the performance of the IIoT network since it utilizes fewer resources. The resources consumed to execute the transactions and device communication are not high. The endorsement peers utilize more resources as they participate in the consensus. Since only a limited number and verified nodes participate in reaching consensus, unlike a public blockchain, our system has low resource usage with high performance.

TABLE V. PERFORMANCE METRICS

Tx Type	Successful Tx	Send Rate (TPS)	Latency (s)			Throughput (TPS)
			Max	Min	Avg	
query	100	7.0	3.50	1.52	2.49	4.4
	400	6.4	3.04	1.12	1.94	5.5
	1300	5.2	2.50	1.03	1.76	4.8
	1600	5.7	3.95	1.01	2.5	5.4
	2500	3.1	1.71	0.8	1.08	2.9
	2800	5.7	2.80	0.90	1.85	4.6
	3700	6.7	3.15	1.10	2.1	5.3
	4000	4.3	1.90	0.85	1.37	3.8
	4900	6.2	2.97	1.01	1.83	4.9

TABLE VI. RESOURCE CONSUMPTION ON AVERAGE

Type	Tx	CPU%(avg)	Memory [MB] (avg)
Docker	100	1.89	76.34
	400	5.32	200.45
	1300	18.67	210.56
	1600	24.33	217.45
	2500	31.23	229.78
	2800	42.55	251.45
	3700	54.55	278.45
	4000	55.89	263.24
	4900	72.33	303.25

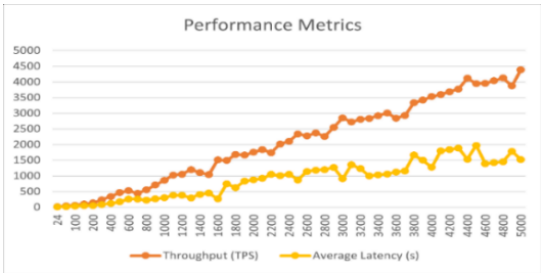

Fig. 8. Performance Metrics.

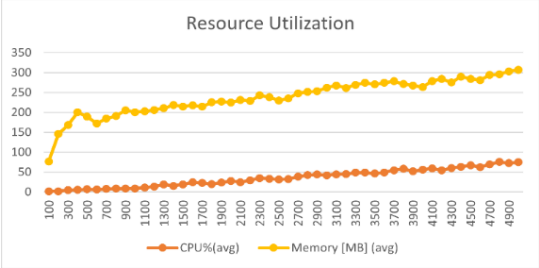

Fig. 9. Resource Utilization.

C. Comparison

The proposed permissioned Hyperledger Fabric Blockchain-enabled IIoT network addresses and improves the security challenges of IIoT as it restricts access and significantly increases throughput, reduces delay in transaction execution, and enhances network performance with required resource usage. There have been studies to use Hyperledger Fabric to improve the security of different industries including [34] where a physical access control management system is developed. Fig. 10 to 15 shows a comparison analysis between using a Hyperledger Fabric for a physical access control device [34] and an IIoT device as our approach. Fig. 10 presents the blockchain network for IIoT devices that have higher throughput than a physical access device [34]. Although [34] performs good with physical control devices, it cannot perform well with IIoT devices in terms of throughput and latency as evident in Fig. 10 and 11. With a lower latency than [34], our approach ensures better performance as it can quickly perform the validation of a transaction. The faster the transaction execution consensus will be received from a limited number of nodes, the more secure the communication medium will be. Therefore, our approach allows low response time with more valid transactions' execution. Fig. 12 presents a combined form of the comparative analysis in terms of throughput and average latency where our approach has a higher throughput and lower latency than [34]. Some of the comparative values of throughput and average latency have been presented in Tables VII and VIII, showing the percentage of increase in throughput and decrease in latency in comparison between IIoT and [34].

However, as presented in Fig. 13 and 14, our approach utilizes more resources than the system of [34]. The authors in [34] have developed an application to control user access through physical devices using Hyperledger Fabric. These physical devices require much less power and space compared to IIoT devices. They are used in certain and specific areas where they are connected to a certain number of other devices.

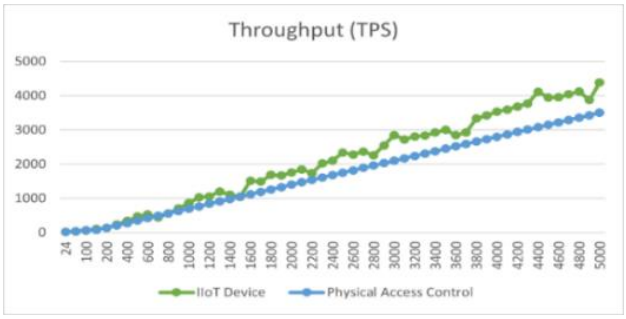

Fig. 10. Comparison of the Performance Metrics: Throughput.

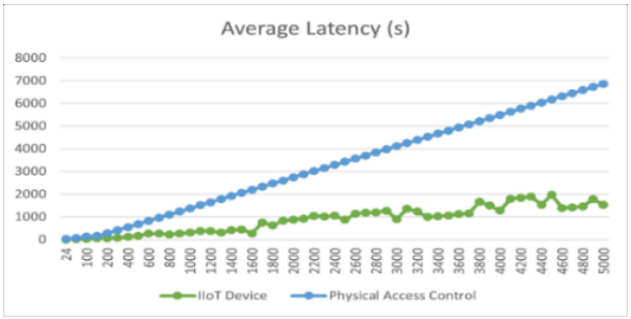

Fig. 11. Comparison of the Performance Metrics: Average Latency.

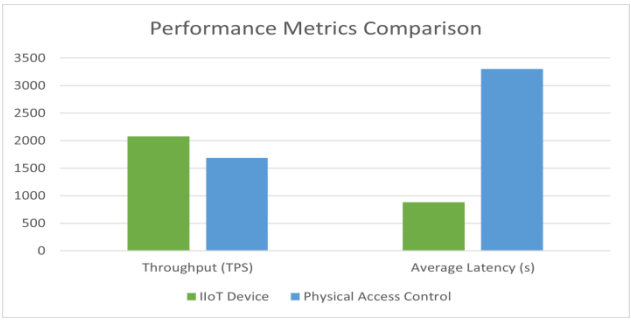

Fig. 12. Comparison of the Performance Metrics.

On the other hand, numerous IIoT devices are used in huge industrial sectors, sharing information and requiring real-time data processing. They need specific storage capability and huge processing power to be able to communicate efficiently [35]. A simple physical access control device merely logs the user access records. In contrast, IIoT devices not only share information but also produce data and provide improved business decision insights, requiring them to comprise with high processing power and memory usage [35]. Therefore, to meet the high performance requirements of IIoT devices [35], our approach uses more resources and secures the network from malicious attackers. It also allows using any consensus algorithm as per IIoT requirement. The algorithm ensures the participation of only a limited number of nodes in reaching a consensus. Consequently, it assures low response time and required usage of resources that helps in having an IIoT permissioned blockchain network with high performance and security. The comparative analysis of the resource utilization by our approach and [34] is highlighted in Fig. 15.

TABLE VII. THROUGHPUT COMPARISON

Throughput (TPS)			
Tx	**IIoT/Our Approach**	**Physical Access Control [34]**	**Performance**
100	62.85	70	10% Decrease in Throughput
400	343.75	280	23% Increase in Throughput
1300	1200	910	32% Increase in Throughput
1600	1515.789	1120	35% Increase in Throughput
2500	2338.71	1750	34% Increase in Throughput
2800	2259.649	1960	15% Increase in Throughput
3700	2926.866	2590	13% Increase in Throughput
4000	3534.884	2800	26% Increase in Throughput
4900	3872.581	3430	13% Increase in Throughput

TABLE VIII. AVERAGE LATENCY COMPARISON

Average Latency (s)			
Tx	**IIoT/Our Approach**	**Physical Access Control [34]**	**Percent Improvement**
100	35.57	137.2	74%
400	121.25	548.8	78%
1300	300	1783.6	83%
1600	267.89	2195.2	88%
2500	870.96	3430	75%
2800	1200	3841.6	69%
3700	1159.70	5076.4	77%
4000	1274.42	5488	77%
4900	1785.65	6722.8	73%

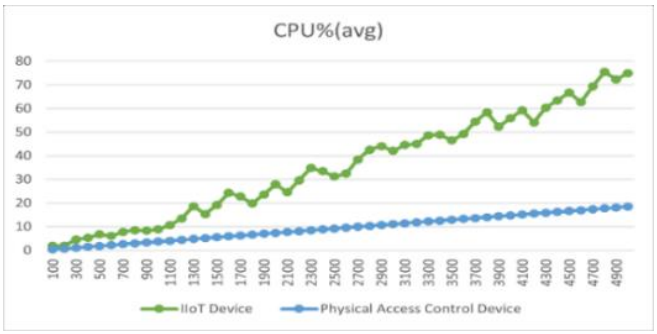

Fig. 13. Comparison of CPU usage.

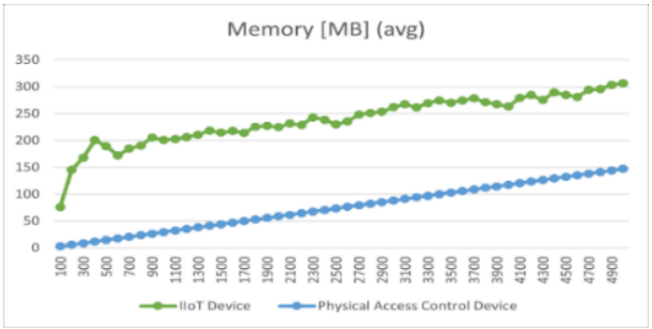

Fig. 14. Comparison of Memory usage.

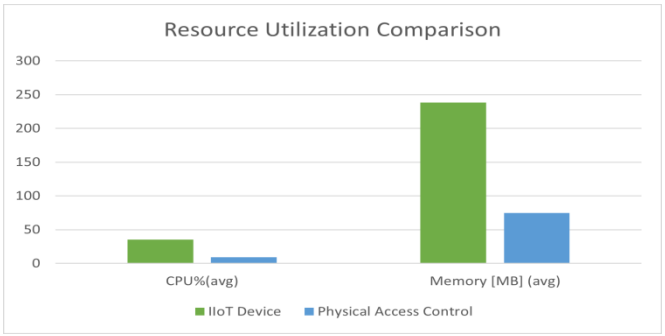

Fig. 15. Comparison of Resource usage.

X. CONCLUSION

With the emerging and diversified use of IIoT, it is important to address the security vulnerabilities of this technology. The integration of blockchain and IIoT can play a vital role in overcoming the IIoT security limitations. Therefore, this paper proposes to use a Hyperledger Fabric Blockchain to secure the IIoT device communication and ensure data is stored, accessed, and monitored by only authorized parties. The Hyperledger Fabric Blockchain-enabled IIoT uses a CA to issue certificates to the identities and authorize them to perform transactions, and MSP defines user access roles based on the certificates. The peers of the blockchain network validate transactions using the definition provided in the Chaincode and generate the transactions as blocks in the network if verified by following Chaincode, and prevent the double-spending attack. It also updates the rest of the peers and provides consistency across the whole network. This paper implements the proposed solution and performs an extensive evaluation in terms of throughput, latency, and resource utilization, to analyze the security of the communication medium. Using the optimum values, the performance analysis indicates that the Hyperledger Fabric blockchain is suitable for the IIoT network that improves the security and ensures only authorized and authenticated identities are participating in device communication.

XI. FUTURE WORK

This study focused on securing the IIoT device communication medium using Hyperledger Fabric and ensuring that security management remains intact. In future research, an extensive study will be performed in guaranteeing the proposal follows the CIA triad and is available to only authorized users. The need for such future work is required to solve and improve the utilization of resources by the IIoT devices. It will also include the study and future analysis on the security vulnerabilities of blockchain that might affect the IIoT environment. Furthermore, future studies will be dedicated to discovering any exploitable bugs in the proposed network.

REFERENCES

[1] S. Yeasmin, A. Baig. "Permissioned Blockchain-based Security for IIoT." In 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), pp. 1-7. IEEE, 2020.

[2] A. Esfahani, G. Mantas, R. Matischek, F.B. Saghezchi, J. Rodriguez, A. Bicaku, S. Maksuti, M.G. Tauber, C. Schmittner, J. Bastos. "A Lightweight Authentication Mechanism for M2M Communications in Industrial IoT Environment". IEEE Internet of Things Journal, vol. 6, no. 1, pp. 288-296, Feb. 2019.

[3] W. Yang, S. Wang, X. Huang, Y. Mu. "On the Security of an Efficient and Robust Certificateless Signature Scheme for IIoT Environments". IEEE Access, vol. 7, pp. 91074-91079, 2019.

[4] Y. Zhang, R. H. Deng, D. Zheng, J. Li, P. Wu and J. Cao, "Efficient and Robust Certificateless Signature for Data Crowdsensing in Cloud-Assisted Industrial IoT," in IEEE Transactions on Industrial Informatics, vol. 15, no. 9, pp. 5099-5108, Sept. 2019, doi: 10.1109/TII.2019.2894108.

[5] J. Huang, L. Kong, G. Chen, M. Wu, X. Liu, P. Zeng. "Towards Secure Industrial IoT: Blockchain System With Credit-Based Consensus Mechanism". IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3680-3689, June 2019.

[6] W. Liang, M. Tang, J. Long, X. Peng, J. Xu, K. Li. "A Secure FaBric Blockchain-Based Data Transmission Technique for Industrial Internet-of-Things". IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3582-3592, June 2019.

[7] Y. Lu, X. Huang, Y. Dai, S. Maharjan, Y. Zhang. "Blockchain and Federated Learning for Privacy-Preserved Data Sharing in Industrial IoT". IEEE Transactions on Industrial Informatics, vol. 16, no. 6, pp. 4177-4186, June 2020.

[8] Q. Wen, Y. Gao, Z. Chen, D. Wu. "A Blockchain-based Data Sharing Scheme in The Supply Chain by IIoT". In 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), Taipei, Taiwan, 2019, pp. 695-700.

[9] J. Wan, J. Li, M. Imran, D. Li, Fazal-e-Amin. "A Blockchain-Based Solution for Enhancing Security and Privacy in Smart Factory". IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3652-3660, June 2019, doi: 10.1109/TII.2019.2894573.

[10] T. Alladi, V. Chamola, R. M. Parizi, K. R. Choo. "Blockchain Applications for Industry 4.0 and Industrial IoT: A Review". IEEE Access, vol. 7, pp. 176935-176951, 2019.

[11] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, M. Gidlund. "Industrial Internet of Things: Challenges, Opportunities, and Directions." IEEE Transactions on Industrial Informatics, vol. 14, no. 11, pp. 4724-4734, Nov. 2018.

[12] P. Sethi, S. R. Sarangi. "Internet of things: architectures, protocols, and applications". Journal of Electrical and Computer Engineering, 2017.

[13] W. Liang, M. Tang, J. Long, X. Peng, J. Xu, K. Li. "A Secure FaBric Blockchain-Based Data Transmission Technique for Industrial Internet-of-Things". IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3582-3592, June 2019, doi: 10.1109/TII.2019.2907092.

[14] S. Yeasmin, A. Baig. "Unblocking the Potential of Blockchain". In 2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA), Ras Al Khaimah, United Arab Emirates, 2019, pp. 1-5.

[15] I. Makhdoom, M. Abolhasan, W. Ni. "Blockchain for IoT: The challenges and a way forward". In ICETE 2018-Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, 2018.

[16] A. Dorri, S. S. Kanhere, R. Jurdak, P. Gauravaram. "LSB: A Lightweight Scalable Blockchain for IoT security and anonymity." Journal of Parallel and Distributed Computing 134 (2019): 180-197.

[17] N. Andola, M. Gogoi, S. Venkatesan, S. Verma. "Vulnerabilities on hyperledger fabric," Pervasive and Mobile Computing, 59, 101050, 2019.

[18] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan. "Hyperledger fabric: A distributed operating system for permissioned blockchains". In Proceedings of the thirteenth EuroSys conference, pp. 1-15. 2018.

[19] Corda, Available online: https://docs.corda.net/key-concepts-notaries.html.

[20] A. Vyas, L. Nadkar, S. Shah. "Critical Connection of Blockchain Development Platforms". In International Journal of Innovative Technology and Exploring Engineering (IJITEE), 2019, ISSN: 2278-3075, Volume-8, Issue- 9S2.

[21] T. Koens, S. King, M. van den Bos, C. van Wijk, A. Koren, "Solutions for the Corda Security and Privacy Trade-off: Having Your Cake and Eating It".

[22] R3 Corda, Available online: https://www.r3.com/corda-platform/.

[23] M. Hearn. "Corda: A distributed ledger". Corda Technical White Paper, 2016.

[24] Openchain, Available online: https://www.openchain.org/.

[25] T. T. A. Dinh, R. Liu, M. Zhang, G. Chen, B. C. Ooi J. Wang. "Untangling Blockchain: A Data Processing View of Blockchain Systems". In IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 7, pp. 1366-1385, 1 July 2018.

[26] D. Schwartz, N. Youngs, A. Britto. "The ripple protocol consensus algorithm", Ripple Labs Inc White Paper, 2014, 5(8).

[27] Ripple, Available online: https://ripple.com/xrp/.

[28] P. Sajana, M. Sindhu, M. Sethumadhavan. "On blockchain applications: hyperledger fabric and Ethereum". International Journal of Pure and Applied Mathematics, 118(18), 2965-2970, 2018.

[29] P. Thakkar, S. Nathan, B. Viswanathan. "Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform". In 2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Milwaukee, WI, 2018, pp. 264-276.

[30] Hyperledger Fabric, Hyperledger Fabric, Available online: https://hyperledger-fabric.readthedocs.io/en/release-2.0 [Accessed on 01 March 2020].

[31] Hyperledger Fabric, 2018, Available online: https://cloud.ibm.com/docs/blockchain?topic=blockchain-hyperledger-fabric [Accessed on 01 March 2020].

[32] Sadeghi, C. Wachsmann, M. Waidner. "Security and privacy challenges in industrial Internet of Things". In 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), San Francisco, CA, 2015, pp. 1-6.

[33] Hyperledger Caliper, Hyperledger Caliper Documentation, Available online: https://github.com/hyperledger/caliper [Accessed on 1 November 2020].

[34] S. Rouhani, V. Pourheidari, R. Deters. "Physical Access Control Management System Based on Permissioned Blockchain". In 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. 1078-1083.

[35] G. Caiza, M. Saeteros, W. Oñate, M. V. Garcia. "Fog computing at industrial level, architecture, latency, energy, and security: A review". Heliyon, 2020, 6(4), e03706.