PRIVACY PRESERVED MODIFIED CONSENSUS MECHANISM FOR BLOCKCHAIN EMPOWERED DECENTRALIZED FEDERATED LEARNING FRAMEWORK IN WIRELESS NETWORKS

By

SYED BASIT ALI ZAIDI

Supervisor

DR. HAIDER ABBAS

A thesis submitted in the Information Security Department, Military College of Signals, National University of Sciences and Technology (NUST), Islamabad, Pakistan in partial fulfilment of the requirements for the degree of MS in Information Security.

FEBURARY, 2022

CERTIFICATE

This is to certify that **NS SYED BASIT ALI ZAIDI** Student of **MSIS-18** Course Reg. No **00000320805** has completed his MS Thesis title "**privacy preserved modified consensus mechanism for blockchain empowered decentralized federated learning framework in wireless networks**" under my supervision. I have reviewed his final thesis copy and I am satisfied with his work.

Thesis Supervisor

(**Dr. Haider Abbas**)

Dated: ______Feburary 2022

Declaration

I hereby declare that no portion of work presented in this thesis has been submitted in support of another award or qualification either at this institution or elsewhere

Dedication

“In the name of Allah, the most Beneficent, the most Merciful"

I dedicate this thesis to my family and to my teachers who guided me through the way.

Acknowledgements

I am most grateful to ALLAH Almighty for all His blessings throughout my master’s degree. Indeed, this would not have been possible without his blessings and guidance through every step. Indeed, none be worthy of praise but Allah. In addition, my respects be upon Prophet Hazrat Muhammad (PBUH) and his Holy Household for being the source of guidance for me.

I would like to pay my greatest regards and my special thanks to my supervisor Dr. Haider Abbas (NUST) for his generous help throughout my thesis, and for truly believing in me, his support and confidence in me drive me to completion of my thesis. In addition, I would also like to extend my sincere regards to all my teachers (Sir Faisal Amjad, Sir Imran Rashid, Sir Yawar Abbas, Sir Waseem Iqbal, Sir Fawad Khan and Sir Shahzaib Tahir) for their support and guidance throughout my course work.

I would also like to thank Prof. Muhammad Imran (University of Glasgow) for his continuous guidance, support and mentorship throughout my master’s degree.

Last, but not the least, I am very grateful to have such supportive and guiding parents. They are always a great source of inspiration to me and are always there for me. I would like to thank them for all their care, love, and support through my times of stress and excitement.

Abstract

Advancements in communication, computing and sensor technology are creating new alternative solutions to the traditional approaches of meeting the ever-growing influx amount of data point demand of 21st century and beyond. To learn and improve on these data points for model training at edge devices, federated learning, as a potential distributed machine learning approach has recently surfaced to leverage multiple nodes to take up some parts of a large training process and aggregate it on a central server. This, however, poses a serious risk towards the privacy of the exchanged data points between the server and the nodes, creating a single point of failure.

Most existing work focuses on how to improve Machine Learning performance without considering security. In this work we focus on to protect the data security without much compromising on Machine Learning performance. To achieve this idea, we will first investigate the consensus mechanisms of Blockchain, followed by an overview of related work and then comparison of Blockchain based Federated Learning framework. If possible, we may also end up proposing a new consensus mechanism to achieve the privacy preserving Federated Learning framework.

1. Explain why we need to integrate the two technologies, blockchain and FL. (For this we need to investigate the benefits and use cases of blockchain in ML/AI.)
2. Illustrate the network/system architecture which achieves the blockchain-enabled FL framework.
3. Compare some existing blockchain consensus mechanisms and give their pros and cons when applying to the above blockchain-enabled FL network.
4. Discuss how the blockchain-enabled FL network can be affected under different types of attacks (e.g., participant in each learning round sends a wrong update information or send the previous value without doing any learning work but getting the benefits (i.e., learning results) from other learning agents, etc.). Conclude which consensus can perform better under different attacks.
5. Conduct simulations to compare the performance of several consensus. We should consider several types of attacks when doing simulations.

Table of Content

LIST OF FIGURES

LIST OF TABLES

Chapter 1

1. INTRODUCTION

1.1 Chapter Outline

In this chapter the introduction of Blockchain Technology and Federated Learning is discussed. The research problem and methodology of how research is conducted are also mentioned in this chapter.

1.2 Blockchain Technology

An exchange of information between two persons can be thought of as a transaction. This transaction is stored on a ledger to record that the event has occurred. Traditionally this ledger is maintained by a central authority, but if this ledger is stored at multiple different locations, it gives a notion of a distributed ledger system. Blockchain technology is also a type of distributed ledger technology in which we have a network of computers that are connected with each other making a peer-to-peer network. Now whenever a transaction is performed it is verified by other users in the network. The validated transactions are then hashed out and grouped into a block, this block is then appended with rest of the blocks such that the previous and current blocks are connected with each other. Since blocks are linked together using hashes and timestamp, there is very less chance that the transaction once stored on the blockchain can be altered. The state of the system is maintained by multiple users by reaching on a consensus. This consensus is reached using different methods. Prominent of which are proof of work and proof of stake consensus mechanism. Among other benefits of blockchain technology, decentralization, reward mechanism and validation of records make it a promising application. [1]

1.3 Federated Learning

Federated learning allows training a machine learning model using data stored at various location without the need to transfer it to a central authority. In traditional machine learning, we require users to send their data to a central authority for model training. This raises some privacy concerns to users. Google in 2016, used the concept of federated learning to train a model by letting users to keep their data at their end. Through federated averaging the central server will receive the updates from the users and then average them out to update the current model. [2]

1.4 Problem Statement

Most existing work focuses on how to improve Machine Learning performance without considering security. In this work we focus on to protect the data security without much compromising on Machine Learning performance. To achieve this idea, we will first investigate the consensus mechanisms of Blockchain, followed by an overview of related work and then comparison of Blockchain based Federated Learning framework.

1.5 Objectives

1. Literature review of how blockchain can be used in federated learning.
2. Comparison on consensus mechanism for Blockchain based federated learning framework.
3. To conduct some simulations to demonstrate this comparison.
4. Discussion on which consensus mechanism could be best for federated learning under different conditions. Use cases of proposed consensus mechanism for federated learning framework and then propose which mechanism is good.

1.6 Research Methodology

The research has been conducted in the following stages:

- Preliminary study.
- Literature survey.
- Explain why we need to integrate the two technologies, blockchain and FL.
- Illustrate the network/system architecture.
- Simulations and result analysis.

1.7 Thesis Outline

- *Chapter 1:* Chapter 1 focuses on Introduction to topic, Problem Statement, and Research Methodology.

- *Chapter 2:* Chapter 2 focuses on preliminary study and the need to integrate the two technologies.

- *Chapter 3:* Chapter 3 focuses on literature review and detailed study of Blockchain and Federated learning.

- *Chapter 4:* Chapter 4 focuses on Proposed Architecture.

- *Chapter 5:* Chapter 5 focuses on Experimental Setup and Result Analysis.

- *Chapter 6:* Chapter 6 focuses on Conclusion, Challenges and Future Work.

1.8 Taxonomy of the Paper

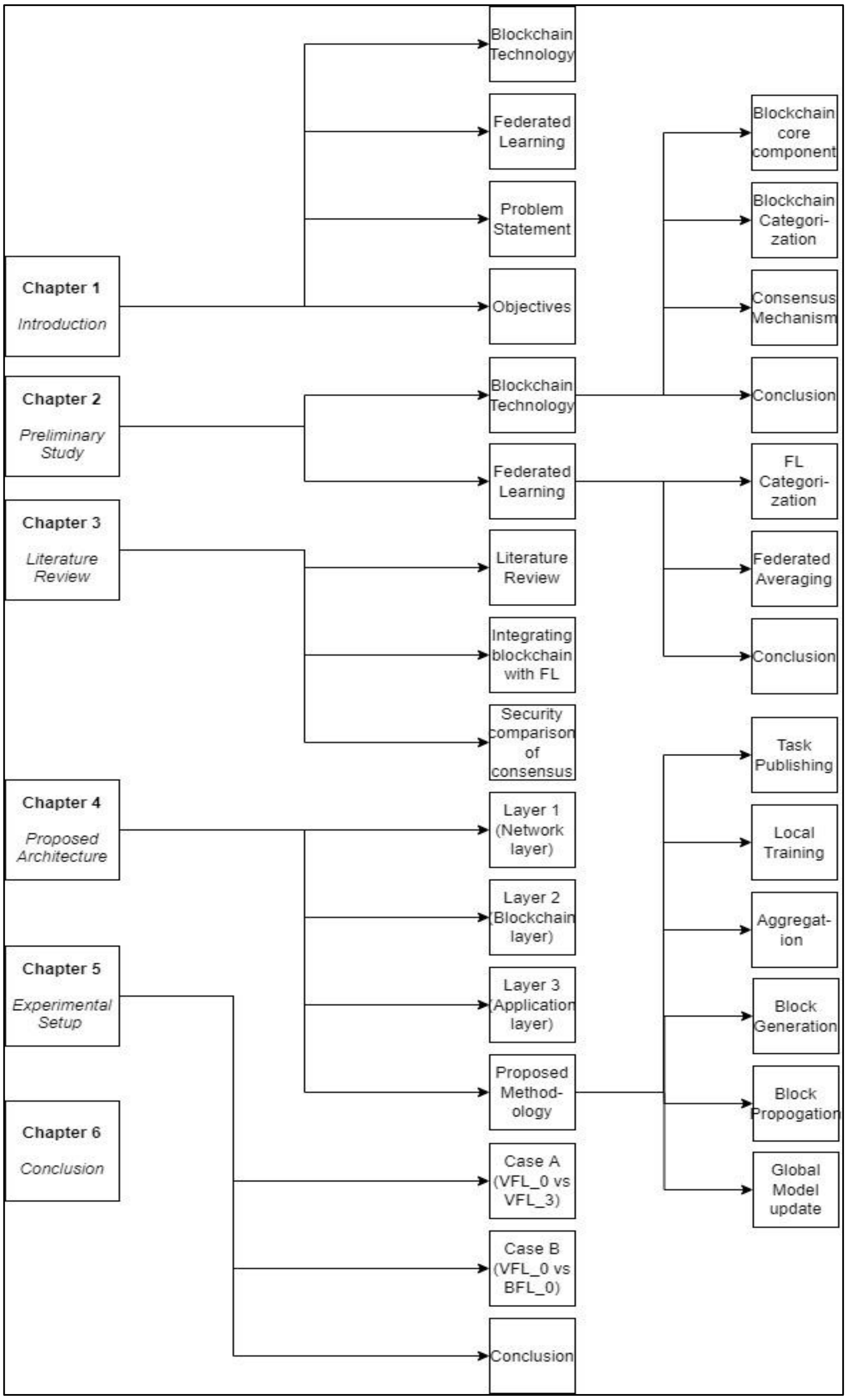


Figure 1.1: Taxonomy of the paper outlines the chapters and sub sections.

Chapter 2

2. PRELIMINARY STUDY

2.1 Chapter Outline

In this chapter we focus on preliminary study about what is blockchain, its core components, concept and features are explained along with this we also detailed about the concepts of federated learning technology, its types, core components.

2.2 Blockchain Technology

Blockchain leverages a community of users to be able to record the information as transactions in digital ledgers in a tamper-proof, non-editable and distributed manner. Thus, blockchains are nothing more than digital ledgers which is distributed among participating peers. 2008 was the year when the term blockchain was coined for the first time. It was for the first time that we saw how a digital currency, Bitcoin, without a central authority was exchanged between distributed peers in a protected way with the help of cryptographic mechanisms. If an organization is planning to implement blockchain in the network, it must consider the fact that the information once stored on the blockchain cannot be deleted or modified, rather a new block is added to the chain, this would keep a full history of the transaction activity. Implementation of blockchain can be both permissioned and permissionless. As the name suggest if anyone can join and read/ write to it, then this type would be called as permissionless blockchain, whereas if the joining peers are restricted to few numbers with controls defined to their roles, then this type is called as permissioned blockchain network. If we examine blockchain, then the body of a block would be broken down in following components; a block header which contains the metadata regarding the block, the data part contains the individual transactions and a hash which would link it to its previous block. If we talk about a transaction, then it would require at-least two peers, one of them would digitally sign it and a record of the activity.

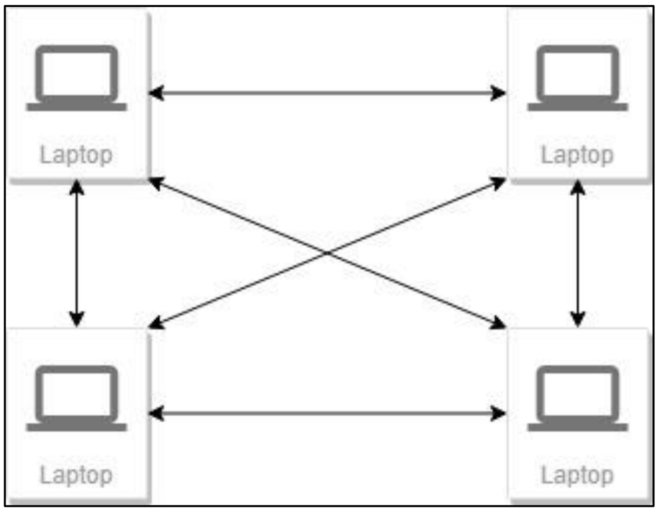


Figure 2.1: Blockchain Peer to Peer Network

2.2.1 Miners

Miners in blockchain are those participants who publish a block and keep a copy of the digital ledger.

2.2.2 Ledger

Blockchain implements a concept of append only ledger, through which any update to the previous record can only be done by appending a new block to the chain, this keeps a track history of all the transactions happened previously

2.2.3 Shared

Since this is a decentralized network, so all the transactional activity is recorded at every peer, this limits the chances of any wrong acts from a central authority, thus providing transparency in the system.

2.2.4 Consensus

If we want a bunch of computers to agree on something, we try to achieve a consensus. Consensus is also called as agreement between parties. In a distributed setting, the need for consensus is important whenever we want a group of computers to agree on the access to a resource, or to make leader among participating nodes, and to agree on the order in which the events in the system would occur.

Table 2.1: Blockchain History

Version	Description	Example
1.0	Digital Crypto Currencies	Bitcoin
2.0	Smart contracts	Ethereum
3.0	Decentralized apps	Opensea

2.2.5 Categorization of Blockchain

Categorization in blockchain depends on who can publish a block, e.g if anyone in the network has the ability to publish the block, it will be called as permissionless categorization whereas on the other hand if only a handful of participants have the leverage to publish the block, then this type of setting will be called as permissioned blockchain categorization.

2.2.6 Permissionless

Such digital ledgers that are free to be joined by anyone and in which anyone has the leverage to publish the block are called as permissionless blockchain network. These are often found to be created as open-source software by public and can be freely used by public as well. Now if users have the leverage to public the block then they also have the leverage to read the blocks. This can lead to some problems in this type of open to join network, and that is the malicious participants may try to harm the network. This problem is solved by employing a concept of consensus mechanism to make participant agree on a final state of the block.

2.2.7 Permissioned

Users who will publish the blocks must be firstly authorized by some entity. In this type we can restrict the read/ write access of the transactions into the block because the authorized peers are maintaining the blockchain network. Like the permissionless blockchain, permissioned blockchain do have some common features, like they can also be made by using open-source software, plus they have the same concept of consensus mechanism. Peers join a permissioned network by giving their identity as authorization step, hence it makes the peers to trust each other in publishing and maintaining the blockchain network. As there is transparency in permissioned blockchain network any malicious activity can be held accountable.

2.2.7 Blockchain Core Components

Now we will investigate some of the core components of the blockchain individually, this will help understand blockchain concept more easily.

2.2.8 Cryptographic Hash

Through hashing what we achieve is that for any given arbitrary input (file, pdf, images, exe, text etc) we get a unique output, which cannot be reversed to find the actual information. This output is called message digest. The unique property of hashing is that if for two identical files as input, we change even a bit in one file then the resultant hash of both files would be drastically different. In blockchain, the common hashing function used is called as SHA-256. Hash function plays important role in blockchain by generating identifiers and to derive address, moreover, the block is made secure by taking out the hash of it. Also the block header contains the hash of previous block to maintain the chain.

2.2.9 Properties of a Hash function

Three main properties of a hash function are listed as;

Hash function are one-way functions; means we cannot calculate the input if we are given the output value. This property is called as pre-image resistant.

e.g for a given message digest, find value of ‘x’

hash(x) = message digest

Hashes of two different files can never be the same. This property is called as second pre-image resistant.

e.g for a given value of x, find value of y

hash(x) = hash(y)

We cannot have two files that results to a same output. This property is called as collision resistant.

e.g find the values of ‘x’ and ‘y’, the hashes of which results to

hash(x) = hash(y)

2.2.10 Nonce

Let us suppose, we are having same input messages, and we need to store them in the blockchain, but also want them to be distinguished, what we do is we apply the concept of nonce. If we use an arbitrary number only once in our input messages, then it is called as a nonce. We add nonce into the input data message so as to change the output message digest. Proof of work uses nonce.

Hashing function (Input data + nonce) = output data

2.2.11 Transactions

In simple terms any sort of contact between participating parties in the blockchain network is recorded as a transaction. There could be scenario when the block does not even contain any transaction, but to maintain the security of the system the blockchain network needs to chain new blocks.

2.2.12 Transaction validity and authenticity

Validity of transaction means it meets some predefined requirements. Whereas Authenticity of transaction means that the participant that initiated the transaction was the person in possession of that data.

2.2.13 Digital Addresses

Blockchain can use addresses as users' identifiers, which can be derived by taking hash of the public key.

2.2.14 Private key Storage

In permissionless blockchain setting there is a need for users to securely manage their private keys. This is done with the help of wallet. A wallet can store multiple information of user (public / private keys and address). Since through private key the blockchain transactions can be performed, if lost, can leverage malicious actors to get hold of the account.

2.2.15 Ledgers

Whenever there was an exchange between two users, tradition was to physically record it on book, thus calling them ledgers. When we apply the same concept in digital domain, we call it as digital ledgers. As the amount of data exchange increased, so does the need to store that record on some database. There are third parties that deploy servers just to store millions of records. This makes users to heavily rely on some central authority. But the concept of distributed setting has made blockchain a key enabler by providing a distributed framework.

2.2.16 Centralized ledger vs decentralized ledger

Table 1.2: Comparison of Centralized vs Distributed Ledger Technology

Serial No.	Centralized Ledger	Decentralized Ledger
1	Ledger can be corrupted/ modified/ lost.	Since blockchain has a distributed setting, the peers in the network maintain a copy of the ledger, thus making multiple backups of the data.
2	If ledger is being maintained on a same network infrastructure as other software, it heavily increases the chances that an attack on the network can potentially bring the whole system down.	The ledger is maintained on different networks, where there are different nodes. So, attack on one node will not hinder the performance of the system.
3	Geographically if the ledger is maintained at one location. Any disruption in that area can bring down the system.	Since blockchain is maintained by peers that are located geographically apart from each other, hence failure of one node won't affect the system.
4	A user must believe that its transactions are being transparently validated.	Peers in the blockchain network, after validation publish a block, hence there is less chance of malicious node to upload an invalid transaction.
5	In central setting, user must trust that the transactions are correct have not been altered.	The ledgers are signed through cryptographic means in blockchain, this makes it tamper proof.
6	It could be that the system on which the ledger is deployed is not getting timely patched.	Peers are distributed in network, if one peer is less security patched, it will not matter.

2.2.17 Proof of Work

The peers try to compete for publishing the block based on who correctly and firstly solve a mathematical problem. The answer to the problem is known to the users, users attempt hit and trial basis to get to the answer, once the user who successfully solve the problem, it will then publish the solution, this will enable other users to validate this solution and the node that solved this problem will publish the block [3].

Example of a computational problem can be any mathematical problem whose solution is difficult to figure out, but as soon as one gets to the solution others can validate it easily. Commonly, guessing the right number of 0's in message digest (hash) is what proof of work tries to implement in its model. As the number of 0's increases or decreases so does the difficulty problem, with more zeroes to guess, the solution becomes hard to guess and vice versa. This level of difficulty ensures security of the blockchain system. The puzzle/ problem is independent for the current block, means if one node finds the solution, it will be validated by others and that node who solved the puzzle will be asked to discard its work and get incentive against it. Hence, to publish new block to the blockchain, there will be another problem to solve

2.2.18 Proof of Stake

Stake can be defined as the amount of assets that an individual holds. So, in blockchain application of cryptocurrency we can say that proof of stake model would use the local cryptocurrency as users stake in the network [4]. The more this stake is invested into the system, the more likely they will want the system to remain stable and will in return have more advantage of publishing new blocks in the system to keep it running and vice versa. Thus, in this model the factor that will determine who will publish next block depends upon the amount of stake the user has invested. So, let's suppose 1st user has invested 60% of the total assets in the system and 2nd user has invested 2% of the assets. Then in this case 1st user would have great leverage in publishing the blocks in system. Publishing of new block is related to the percentage of users stake to overall amount staked in the blockchain network. Unlike PoW, users do not have to exhaust their computational power/ resources, rather they invest their assets. Now because the users are not putting in any computational work, they are not incentivized the same way users are in PoW, rather all the cryptocurrency is already with the users themselves and the amount of currency they have invested into the system. PoS is implemented in four different ways.

2.2.19 Random Selection (Chain-based PoS)

In this type of setting, the network will look at all the participants and will decide among the participants that who has the highest stake. The participant with the highest stake among all other participants will be the one to publish the block every time. Greater stake leads to greater chances of being selected as a publishing node.

2.2.20 Multi-round Voting System (Byzantine Fault Tolerance PoS)

In this type of setting, the network will gather all the participants and will perform voting among them, the participant with the highest votes will get the chance to publish the block.

2.2.21 Coin-age PoS

In this type of setting, the asset involved has an age related to it. And after a set time period has passed the assets will have its age reset and that asset will not be used again.

2.2.22 Delegate System

This type of setting is a sense of some combination of both chain based and byzantine fault tolerance. The participants in this type of setting will cast their votes and the voting power (weight of the vote) is related to the amount of stake they have put in. So participants that have great stake involved will have more weight to their votes. Participant that gets the highest vote will become the publish the block. This voting can either happen to make or break a publishing participant. Means participants can vote to make a publishing node, and they can also devote to remove that publishing participant.

2.2.23 Round Robin Consensus Mechanism

In this type, the participating parties take turns to publish blocks. Round robin scheduling concept is not new and has been around quite some time. [5] What happens in round robin is that we go from top to bottom in a list in a periodic manner. Now a problem arises in this situation, if the participating party is not available at his turn, then there will be time limit attached to it, if that node gets available in that time period, then it gets its turn to publish the block, else the next node in the round will publish the block. It is a straightforward approach and there must be a trust among the participants wanted to implement this type of consensus. Thus, this type of model is used in permissioned networks. So, if it were applied in a permissionless network, what will happen is that malicious nodes would continuously create and publish newer blocks and taking over the blockchain network.

2.2.24 Proof of Authority

In this type of model can only be implemented in permissioned blockchain, the participating parties must give away some sort of their identity to be part of the network. The idea is that if the nodes have given some real identity to the peers, then it will refrain them to do anything malicious because their real identity is at stake.

2.2.25 Proof of Elapsed Time

In this type of model, the participating nodes need to wait for a random time period, which is given by and after that time period has passed, they can publish the block.

2.2.26 Conclusion

Blockchain provides a means to perform transactions in a distributed fashion, without relying on any centralized entity. The ability to record a chain of events in a distributed fashion and which is not easily tampered with has attracted industry from every vertical to explore and exploit the application of blockchain in their respected domains. This is backed by the fact of recent growth and acceptance of bitcoin as an application of blockchain in financial sector. Data once written on blockchain will always remain there

2.3 Federated Learning

The information exchange between internet connected devices generates great amount of data on daily basis, but there are two sides of a coin [6]. On one side, where it is being utilized to create better application for the enhancement of user experience, it is on the other side raises great concerns among users about the potential misuse/ leakage of their personal data [7-9]. Moreover, countries are also implementing strict user privacy and security laws, prominent of them is European Union’s General Data Protections Regulation (GDPR) law [10]. Similarly, [11,12] China has also devised a Cyber Security Law for the protection of its users in digital domain. To continue with the mantra of user generating ample amount of data to be used in fine tuning the applications, there is much need for a solution which can address the challenge of user privacy and security of data.

Federated Learning Categorization

2.3.1 Horizontal Federated Learning

According to [13], we can divide the data into two major categorizes, sample-based dataset and feature based dataset. Now if data from different users overlap based on feature-set of datasets, then this categorization of federated learning based on feature sample space is called as horizontal federated learning. Now in federated setting, since we have distributed users, hence there will be different datasets, let us suppose we denote dataset of two users as Di and Dj, and we denote feature space as Xi and Xj, and we denote data label as Yi and Yj, then we can assume that the feature and label space of both datasets will be same but the sample space (identifiers) are different.

2.3.2 Security of Horizontal FL

When defining the threat model for a machine learning system, we assume some roles of the participating parties (both users and the server). In Horizontal federated learning, the same threat model concept can be applied where we can assume that the users are honest in nature and the server is also honest in nature but is curious to know the user data, and through this way the server can compromise the user data privacy [14, 15]. Recently, there has been some studies to protect the user privacy for data, prominent work includes that of Bonawitz [15], who proposed a framework for securing the model updates from users by performing model aggregation in a secure manner. Other works of Phong [17], successfully prevented server to investigate the model parameters by applying cryptographic means such as Homomorphic Encryption to securely aggregate the user model parameters. Hitaj [18] work proposed that the model parameters and the aggregated model must be shared with every participant.

2.3.3 Architecture of Horizontal FL

In computer science, we have majorly two distinct type of network/ system architecture. The client-server and the peer-to-peer system architecture.

2.3.4 Client-Server Architecture

In federated learning the client server architecture which work together to build a model, we have N number of participants, now these users will all have same data structure. Like it was mentioned earlier in previous section, that we assume in this type of setting that the users are honest in nature whereas the server is also honest in nature but is also curious to know about the data structure. Now in this type of setting, we need to take care of the fact about server trying to get knowledge of the user data, means to prevent the information leakage.

2.3.5 Peer-to-Peer Architecture

In peer-to-peer network architecture we do not have a central authority dependency. In this distributed setting the N participants goal is to train the same ML model by utilizing their local data.

Threat model for a peer-to-peer Horizontal FL has a requirement to perform secure exchange between the peers and for that public key cryptography can be applied.

Since the network architecture is a distributed network there comes a problem where different peers need to reach to a consensus where they agree on the order of sending and receiving the model parameters. For this we have two approaches.

- **Transferring model parameters in Cyclic order**

The order of transfer is in a cyclic fashion such that peer 1 will send its updates to peer 2 and peer 2 to peer 3……, from peer N-1 to peer N. Then from peer N back to peer 1.

- **Transferring model parameters Randomly**

Let us suppose we have N peers, the Nth peer would select a random number based on the equation as $\{1, 2, 3, \ldots, N\} \setminus N$. The number which would come at random would become the peer to receive the model parameters. Now the selected peer would update the parameters and would in same fashion as 1st peer would select a new peer randomly and send the parameters to him.

With an advantage of removal of central entity in peer-to-peer network architecture, we have a drawback in this, the drawback is that it takes much more time to train the model.

2.3.6 Global Model Evaluation

Since we have a distributed setting and in federated learning the local data is present at individual's device and never shared, hence, the evaluation is done at everyone locally. Problem is that the evaluation can be done only for local model, but how do we do evaluation of global model?

- **Global Model Evaluation in Client-Server Architecture**

  Both the entities need to reach a consensus for global model evaluation. This is explained through classification example in ML. Classification problem can have two possible outcomes for a given input e.g spam and not spam emails. And in this we can be either true or false positive or negative.

  1. Each peer would calculate the evaluation of their model locally.
  2. The results from each peer would then be sent to server.
  3. Server will evaluate global model based on these results from peers.
  4. Server resends the results to the peers.

- **Global Model Evaluation in Peer-to-peer Architecture**

  Lack of central authority makes it difficult to calculate the evaluation of global model. Making nth peer to act as server and then iterate the steps of a client server architecture can be one method of evaluating a global model in peer-to-peer architecture.

2.3.7 The Federated Averaging Algorithm

Client Server FedAvg Algorithm

We firstly initialize a global model, after this in each round the server will send this global model to some of the peers in the network. The peers would then perform the local training using local data and will generate a local model, they would then send the updated parameters to the server again. The server would then aggregate and average out the parameters it has received from the peers and will make a new updated global model [16, 19, 20]

Hyperparameters in P2P FedAvg

There are four hyperparameters in client server based FedAvg, these are as follow

- Number of clients to participate in each round
- Minibatch size
- No of training rounds i.e epochs
- Learning rate

Peer-to-peer FedAvg Algorithm

Previously we investigated how McMahan proposed FedAvg algorithm in a client server architecture, but there is a need to apply FedAvg in peer-to-peer network as well. Most recently, in [16], the authors described how FedAvg can be employed in a peer-to-peer setting. The reliance on central server is removed. In a peer-to-peer network for federated averaging, the peers can directly perform exchanges with each other. Following are the steps.

Peers are represented as ‘C’, Weights as ‘W’, Training rounds as ‘T’, user Data as ‘P’ and model as ‘M’.

- Peers have their respective models.
- Same weight Wo is given to the participants models.
- Every peer train the model using its data present locally. This generates a local model M.
- Every peer then aggregates and average the updates from other peers.
- Now local model is updated

Hyperparameters in P2P FedAvg

There are four hyperparameters in P2P FedAvg, these are as follow

- Neighbors of Nth Peer
- Minibatch size of Local Peer
- Number of times each peer will train using local dataset i.e epochs
- Learning Rate

2.3.8 Vertical Federated Learning

When the participating party’s dataset have similar sample space and they are different from feature set, we categorize it to be as Vertical Federated Learning. In data matrix, the columns represent features of the sample.

Since the data is coming from different users, this means that they have different features. As compared to Horizontal federated learning where the identifier of the users were not similar along with same feature and labels, in Vertical federated learning we have the identifiers of the users to be similar whereas the feature and labels are different

3.3.9 Security of Vertical FL

We assume that the peers in Vertical Fl are honest but are curious in nature, the same we discussed about the server role in horizontal FL. Further to this we get the concept of a third party which is semi honest in nature and who’s duty is to accept the results to calculate the loss function and then it resends this to the peers.

2.3.10 Architecture of Vertical FL

Let us suppose there are two honest but curious organizations that needs to train a machine learning model with their local dataset. Since they cannot share their data with each other, to protect the confidentiality of the data, an honest third organization can be used

Vertical FL Training Process

It is a two-step process. It brings the collaborators on same page by sharing their identities. Then the training starts.

- By using cryptographic encryption techniques for user identity, we bring out the common users from two organizations, without exposing their data.
- The common users would then start training. Further the training process is consisted of four steps.
    1. Honest third party generates public key pairs for common users of two organizations.
    2. In the training process, the users train the local model and share the encrypted results with each other.
    3. Both users send the encrypted results to honest third party.
    4. Third party resend the updated gradients to both users

2.3.11 Conclusion

Federated learning allows multiple parties to hold their own data privately while building a machine learning model collaboratively and securely. With federated learning, data does not need to leave the data owners, and hence privacy can be better protected. In this book, we have discussed several modes in building the federated machine learning model, including horizontal federated learning, vertical federated learning, and federated transfer learning

Chapter 3

3 LITERATURE REVIEW

3.1 Chapter Outline

In this chapter we focus on relevant research work carried out in this domain. Also, we explained why there is a need to integrate the two technologies. We then explained in detailed the security aspect of blockchain consensus mechanisms.

3.1.1 Literature Review

Advancement in mobile phones with multiple sensor technologies on it has leveraged them to become the primary source of computing, generating ample amount of data, most of which is sensitive and private in nature. These great amount of data points could potentially be used as a source to train model which could in turn improve the user experience such as G-board from Google, improved speech recognition, selection of good photos from trained image models. But the risk associated with these data points being stored on a centralized location and training using conventional approaches is of great concern [21].

The concept of Federated learning was firstly introduced by Google [22]. They proposed federated averaging (FedAvg). The main objective of the authors was to come up with a solution where the model is trained in a distributed fashion while preserving the user's data as the data is going to be present on the user device and only the model was shared. To enhance the distributed machine learning paradigm, significant amount of work has been carried out, the work majorly focused on improving the communication rounds and minimize the cost associated while maintaining the efficiency of the framework [23]. In [24], authors also tried to minimize the communication costs by introducing optimization method. Furthermore, in [25], the authors propose FL algorithm for mobile devices. As stated above the mentioned works depends upon a central server, on which local model updates from the participating nodes are aggregated [22].

In [26], the authors highlighted the security aspects of using blockchain in a decentralized fashion.

With large amount of data points transferred between edge devices and central nodes for improved machine learning models in conventional ML schemes comes with undesired delay in the transmission due to poor networking capabilities at the edge devices as well as the privacy issues related to personal data. To resolve this problem an effort to merge both computation power and communication, while ensuring privacy of the raw data to train the model is seen as way towards a privacy preserved federated machine learning concept. Security of global model at the central server, untrustworthy participating nodes in the framework, latency issues arising due to massive number of participating nodes thus reducing efficiency and reward for the participating nodes who are willing to contribute to maintaining the accuracy of the trained model are some the factors which play a key motivation in fusing FL and Blockchain [27].

As an application of Blockchain, bitcoin emerged as a type of decentralized, provenance preserving, and immutable ledger technology where every participating end user can be identified uniquely [27, 28]. Due to inherent characteristics of being tamper-proof, leveraging nodes to be anonymous, and reduced traceability, blockchain in widely being considered as a security feature for IoT devices [29, 30], some of the major application of which are for IoV networks [31] and smart grid [32]. To enable trustworthy record keeping among nodes a decentralized consensus mechanism [33], tamper-proof records [34], and smart contract which gives incentives to nodes in a blockchain network can be merged with FL framework to solve the issues for FL implementation at resource constrained edge devices.

In [35], the authors implement a FL architecture which they named as Block-FL, in this framework the local model updates from distributed nodes are securely exchanged via the blockchain technology. When the model has been trained locally by participating nodes with their local data, it is uploaded to the blockchain and from there the central aggregator after aggregating the global model distributes back to the nodes, this iteratively keeps on going until the global model attains the desired accuracy. The drawback of this framework was reflected in the incentive phase, where the incentive was proportional to the size of data thus not reducing the fraud caused by the malicious nodes in the chain. The authors analyze the latency in the system while optimally generating blocks to demonstrate scalability and robustness.

In [36], the authors proposed another blockchain empowered FL framework which they named as “FL-chain”, the paper was focused on the security in the mobile network. By introducing a concept of “the global model state trie”, the authors were able to form a MEC-enabled blockchain. The global model was calculated by collecting local model updates from mobile devices. In the paper, the authors lacked to address the trustworthiness of local participating nodes.

In [27], the authors proposed a two-layer blockchain architecture, where the authors considered both the local devices and the MEC nodes. The proposed framework considered two types of blockchain which they called as Local model update chain and global model update chain. LMUC was used to store the local devices updates from good reputation MEC nodes, whereas the GMUC contained all the MEC nodes. Thus, increasing the overall trustworthiness and efficiency of the mobile devices in the network. Local model updates were recorded in the blocks in LMUC, in this a D2D communication link was established to quickly process the transmission part exhibited because of consensus. The concept of smart contract was used to reward the devices which are participating in FL framework. This concept which was introduced in this paper provides both trust and efficiency. Block in LMUC records the local model updates and D2D was introduces to tackle with the delay in the consensus process. GMUC obtains the global model updates thus mitigating the malicious local devices. The authors lacked to consider the mobility of local devices in the LMUC phase along with this a comprehensive network attack model was also not included.

Recent studies [38, 39, 40], have demonstrated a trade-off between communication cost and computation, e.g., through compression of local model, periodic averaging, partial participation of mobile devices etc., however, it all results in inaccuracy of global model. Moreover, most studies in federated learning assign weights and averaging to update the global model, however, a recent study [37] has used particle swarm optimization technique to improve the communication performance.

Due to the bottleneck, this approach has recently been addressed by alleviating the role of central server and extending classical consensus mechanism of blockchain into the network, which serves as a theoretical foundation for a blockchain enabled federated learning framework. Combining blockchain and federated learning convex towards a more secured and intelligent data sharing paradigm. Although recent work has replaced the central server by emphasizing on a more decentralized approach, but they inevitably introduced the reliance on a central server e.g., in storage optimization. In addition to this, frequency has been a key metric in federated learning framework literature but lacks to address the performance of federated learning in resource allocation, wireless bandwidth, and client computing resources.

Furthermore, it also lacks to address the management of historical blocks in resource constrained edge devices in the network. Privacy concerns, security attack vectors on the information exchange link, computation cost, delays in the consensus mechanisms and most importantly communication overhead when clients are resource constrained are also not discussed in detail.

In [41], the author has considered addressing the traditional FL framework in which one server was in control of the training process, client management and aggregation of model. After the end of each iteration the updated model is broadcasted to the participating nodes. The nodes use this model to further increase the accuracy with their local data. The server aggregates the global model. The author has proposed Smart contract in blockchain to replace this central server. Competition based and communication-based mechanisms has been used to append the blocks on the chain and to reach an agreement, respectively. In this way only the qualified updates are chained.

3.1.2 Comprehensive Comparison of Literature Review

Table 3.1: Taxonomy of Blockchain based Federated Learning

Classification	Ref.	Consensus Mechanism	Contributions of Paper	Shortcomings
Security and Privacy	[26], [50]	Proof of Work	The authors in two papers proposed mechanism to counter the poisoning attacks.	In experimental section, there was no comprehensive attack simulations.
	[51]	Proof of Work	The authors proposed a multiparty learning framework to protect privacy.	The authors did not evaluate the proposed framework

	[52]	Proof of Stake	The authors proposed a solution to send model updates in a privacy preserved way.	No experimental simulations to demonstrate attacks.
	[53]	Proof of Stake	The authors proposed a blockchain based FL architecture to preserve privacy	Communication cost by preserving privacy in architecture was not discussed.
Communication	[23], [24], [37], [38], [39], [40], [54]	Proof of Work	The authors proposed communication cost analysis for blockchain based FL	Incentive mechanism and security have not been discussed.
	[55]	Proof of Work	The authors discussed communication cost for Internet of Vehicles.	Not compared with vanilla FL.
	[27], [56]	Proof of Stake	Communication cost analysis for MEC devices.	Not compared with PoW consensus mechanism.
Resource utilization	[57]	Proof of Work	Author discussed about resource utilization with Deep	Authors did not consider resource utilization in block mining process.

			Reinforcement Learning approach.	
	[58]	_	Author discussed about resource utilization with DRL approach for sending global model	Authors did not discuss how blockchain is being used in FL.
	[59]	Proof of Stake	Author discussed how to improve resource utilization with DRL approach	Result analysis after using blockchain in FL is not discussed.
Reward Mechanism	[60]	Proof of Work	Authors proposed a rewarding design.	No experimental simulations to show the working of blockchain FL
	[61]	_	Authors proposed a reputation system.	No experimental simulations to show the working of blockchain FL
	[62]	Hyperledger Fabric	Authors proposed a rewarding protocol	No experimental simulations to show the performance

3.1.3 Integrating Blockchain with Federated Learning

To improve the performance of a system we need data that most closely resemble the environment that the system will work in. This data can be collected from many different sources. Our mobile phones are a good source of data to the marketing and ads industry. They can use this data to better understand our living styles and sell their product to us. This collection of data is tolerable, but there are some situations where the owner of the data is not comfortable in giving out their personal data. Moreover, recently countries like Europe have started to implement GDPR (General Data Protection Regulation) to better protect their citizens data. Now in this situation how can industry be able to propose solutions that will benefit its consumers, when they are not able to gather personal data. Similarly, traditional machine leaning models also require great amount of data to be trained over.

There are situations when users are not willing to share their data or in some situations the researcher himself is lacking with enough data to train his model. In traditional machine learning what we see is that users will send the data to the central server and there the training process begins. This concept can face questions particularly when users are aware of rules such as GDPR. To tackle this hurdle, what google did is that they came up with a framework that will leverage users of the learning process to learn the model on their own with their own data locally and then share with central server whatever they have learnt, and the central server would take up all the learnt models from the users ang aggregate it to make a single global model. This can be done iteratively to reach a good accuracy level. Google also applied the concept in Gboard.

The concept of distributed learning was coined. But it still needs some more improvements. Improvements in terms of the central server it employs to aggregate the leant models from the participating users. What if that central server acts maliciously or becomes unavailable? One possible solution is to somehow remove this central server dependency. In recent studies a decentralized system, blockchain, is seen to be employed. Since both the architecture aims towards decentralization concept, it can be accepted that integrating blockchain with federated learning will remove the central entity.

In federated learning system, models from participants are sent to the central server where aggregation is performed. It is possible that due to large number of devices involved in this type of setting, the network resources are consumed heavily. Other situation could be that the server acts maliciously, or the participants starts acting maliciously by sending falsified data.

It could also be that the server faces single point of failure. Therefore, federated learning system must be such that it is stable and resist single point of failure. Since blockchain in decentralized, the idea of system stability is acceptable. Federated learning system must also be such that it is fair and can be trusted by participants. Since blocks in blockchain are stored at every participant, the concept of fairness can be solved.

The massive amount of data being generated by internet connected devices is being utilized to train a machine learning model and make better applications. However, traditional machine learning requires participants to send their personal data to a central server for training purposes. People do not like the concept of sending this personal data to a central entity. For this google proposed a framework named as federated learning, in this framework, participants are not required to send their personal data to a central server, rather the participants only upload the parameters of the trained model from their own data. Through this concept, it is possible to prevent the leakage of personal data. However, there are other prevailing issues with federated learning framework.

- **Central Aggregator prone to single point of failure**

  In federated learning setting, for local model updates aggregation we are bound to rely on a central server. This can pose some issue, where there is a possibility that the central server is not always available, it may act maliciously in aggregating the results or is prone towards external attacks.
- **Malicious Participants sending corrupted data**

  Ideally, we want that all the participants involved are honest and are working towards the betterment of the system. However, in a distributed setting where we have many participants, there may arise issues where the participants are not honest. Rather, they act in a malicious way and upload falsified and misleading data to hinder the performance of the system.
- **No rewarding mechanism**

  Since federated learning requires participants to train the models using their computing resources with their own dataset, there is a possibility that participants are not attracted enough in remaining in the system for long and that too without receiving any rewards for the training work they have performed.

Recent works have suggested Blockchain as a potential candidate to tackle above mentioned challenges in using federated learning system.

Properties like decentralization can tackle with the situation of reliability of central aggregator, where it can be replaced with a peer-to-peer blockchain network and model aggregation can be done by distributed participants and thus reducing the chance of single point of failure in system [42]. Through blockchain transaction validation can be done. This can potentially prevent malicious participants to send falsified data. Through blockchain participants can be incentivized and this can lead to the participants to remain in the learning rounds.

3.1.4 Comparison between Vanilla Federated Learning and Blockchain based Federated Learning

Table 2.2: Comparing Vanilla Federated Learning with Blockchain Based Federated Learning

Classification	**Advantages**	**Shortcomings**	**Working Environment**
Vanilla Federated Learning	Training is carried out without needed to share data.	Due to central server, it is prone to attacks such as single point of failure, malicious participants.	Participants need to trust server.
	Ensure privacy of data.	Model is aggregated without any validation.	
	Lesser usage of networking resources.	No reward mechanism.	

Blockchain Based Federated Learning	Blockchain replaces dependency on central server.	Greater latency.	Training can be done with less computing powered devices.
	Less chances to single point of failure.		

3.1.5 Comparison of some existing blockchain consensus mechanism and give their pros and cons when applying to the above blockchain enabled FL.

A. **Security analysis of proof of work:**

Pros:

1. If the honest nodes can hold greater than 51% of the computation power than the malicious participants won't be able to make their blocks.

2. Proof of work is not vulnerable towards long range attack and coin age attack.

3. Proof of work gives greatest reliability and fairness.

4. Proof of work is fault tolerant.

5. Proof of work is not prone towards 1/3rd byzantine fault tolerant participants.

6. Proof of work also tackles the double spending problem, because to double spend a previous block it has to re-mine all the previous blocks quickly and surpass the legitimate blockchain, this is unlikely because the legitimate chain continues to grow.

Cons:

1. Proof of work needs great amount of energy.

2. Slower than proof of stake.

3. Proof of work is vulnerable towards denial-of-service attacks, selfish mining attacks and sybil attacks.

Vulnerabilities in proof of work:

1. Tradeoff between security and performance
   Proof of work has a throughput with low rate of transaction e.g in the case of bitcoin, the transaction per second is only 7. 10 minute is the block interval duration. If this duration is reduced or increased, it may lead to forking events or a delay in transaction and it can lead to security issues. Security in proof of work is done through hashing.

2. Energy inefficiency
   The amount of energy required for a transaction is 431 kilo watt hour. As the mining difficulty is increased the need of energy to mine a block will increase.

3. Eclipse attack
   The ability of a participant to compute the hash in proof of work establishes its security. In a poorly connected peer to peer network eclipse attack is possible.

4. Selfish mining
   Ideally the concept in proof of work block propagation is that the mined block must be quickly broadcasted. But if the malicious participant retains its mined block and then later publish them it can lead to selfish mining attack. Both eclipse and selfish mining can be assumed to disrupt small blockchains and not older and longer blockchain

5. Centralization
   The reward that a participant gets in proof of work depends upon the amount of computing power it withholds. This is evident with an example that in a blockchain network with participants backed by companies can invest in computation power as compared to individuals.

B. Security analysis of proof of stake:

Pros:

1. Proof of stake consensus mechanism is energy efficient as compared to proof of work.

2. Proof of stake consensus mechanism has token/ votes that tackles the Sybil attacks on the system.

3. Higher the stake, higher is the probability of generating new block, this process of generating new block utilizes much lesser amount of computation power as compared to proof of work.

4. We do not have to perform real mining, rather we have the concept of validators.

5. 50% of tolerance level is achieved in chain-based proof of stake.

6. Committee based proof of stake makes use of stake and assigns orderly turns to publish blocks.

7. The time of publishing next block and participants of the committee can be adjusted so that every participant has received the broadcast message. Example is of cardano crypto.

8. 1/3rd byzantine validators among all the participants in Proof of stake can be taken care.

Cons:

1. In case the malicious participants get their stakes to reach 50% there is a chance that they can carry out double spending attack and publish new blocks. This is like proof of work 51% attack. This can be tackled by devoting the malicious participants and not giving them stakes.

2. To generate new blocks, chain-based proof of stake uses similar hashing mechanism.

3. Committee based proof of stake is vulnerable to 51% attack. Means only if the malicious participants are less than 51%, the good participants can publish the blocks.

4. In committee-based proof of stake if the number of participants increases it will lesser the network performance and increases the communication cost, because it is evident that the greater the number of participants, it would take greater time to come to a consensus.

5. Round robin-based proof of stake is also prone towards scalability.

6. Time adjustment to receive broadcast messages leads towards delays in transactions and results in lesser network/ transaction throughput.

7. If more than 1/3rd tokens are maliciously owned it can corrupt the consensus process.

Vulnerabilities in proof of stake consensus mechanism:

1. Costless attack

   As in proof of stake computation is not required as in proof of work, malicious participants can leverage this to create a side blockchain. Following four vulnerabilities are subcategory of costless attack.

   a. Nothing at stake

      Chain based proof of stake are prone towards double spending attack and can be tackled by charging the malicious participants. The miner can validate multiple transactions and can publish blocks.

   b. Corruption attack

      As the stakes and address can be seen publicly by the participants, malicious participants can grow a side chain and reward the greater stakeholders.

   c. Long range attack

      In long range attack it is assumed that some malicious participants can grow a side chain and become longer than the main chain. Because of lesser stakeholders this is possible that a side chain can be made quickly by doing all the proof of stake blocks.

2. Centralization

Like proof of work, proof of stake is also vulnerable to centralization of stake. Meaning greater stake participant can become a central authority by owning more than 50% of the total stakes and can supersede other participants in publishing blocks.

C. **Security analysis of practical byzantine fault tolerant:**

Pros: [49]

1. The consensus mechanism can withstand n/3rd byzantine nodes when we have n participants.

2. It is energy efficient because it does not involve computation like proof of work.

3. No need for validation of transaction from every participant.

4. Every participant is rewarded.

Cons:

1. If malicious participants are more than 1/3rd the consensus will fail because the good participants won’t be able to conclude on same value.

2. Identity of the participants are known to each other.

3. Prone towards sybil attack (one participant can take over other participants)

4. Scalability issue is seen in practical byzantine fault tolerant consensus mechanism because as the participants number increases it will also increase the time to respond to the request.

5. Can work efficiently when participants are less.

Table 3.3: Comparison of Different Types of Consensus Algorithms

Classification	**Proof of Work**	**Proof of Stake**	**Delegated Proof of Stake**	**Proof of Elapsed Time**	**PBFT**

Blockchain Type	Public	Public & Pvt	Public	Private	Private
Computation	High	Medium	Medium	Low	Low
Network utilization	Low	Low	N/A	Low	High
Malicious activity Tolerance	<25%	<51%	<51%	N/A	<33%
Decentralization	High	High	Medium	Medium	Medium
Latency	High	Medium	Medium	Low	Low

Chapter 4

Proposed Architecture

4.1 Chapter Outline

In this chapter we outline the architecture which achieves the integration of Blockchain with Federated Learning system.

The architecture proposes an integration of blockchain and federated learning to build a blockchain based federated learning framework. The major concern in traditional FL was during the exchange of model parameters between the client and server, we aim to target this area, by decentralizing the parameter exchange at each participating party.

We propose a three-layered blockchain based federated learning architecture.

Layer 1: Network Layer

The network architecture removes the reliance of a central authority by establishing a peer-to-peer network, this layer consists of the participating parties assuming the roles of task publisher and performing ML model training. We assume that the participating parties have their local dataset with them, and all wants to train the same ML model. The aim is to fully decentralize the network architecture, and this is done by leveraging users to train the ML at their end then each participant would query other participants for their results for that round, each participant would then aggregate the results and take out the maximum value from it. This ensures that each participant will know about the results calculated by other participants. Participants would then also mine the blocks.

Layer 2: Blockchain Layer

Blockchain layer is used as a distributed database for the verification purposes, the participating parties will publish the results after the iteration, this includes the aggregation results and the maximum value calculated.

Layer 3: Application Layer

In the application layer, the parties will perform the actual Federated learning process and we can also employ a smart contract.

4.2 Proposed Methodology

Following is the methodology though which our proposed framework would work.

Step 1: Task Publishing to Participants

Firstly, the participants are given the task to perform.

Step 2: Local Training by participants and Parameter Broadcast

Participants will perform training locally using their data. They will then broadcast the results in the peer-to-peer network.

Step 3: Aggregation of updates

When participating parties have broadcasted their respected results to other parties in due time, they will then aggregate the result and update the global model.

Step 4: Participants generating Block

After the iteration is completed, the training participants now become the mining participants, they will begin mining the block. As previously stated, we aim to store the learnt parameters from each party on the block as well. After successful mining of block, other party members will verify it.

Step 5: Propagating the block

After verification step is completed, the block will now be linked in to the blockchain.

Step 6: Global Model is updated by each participant

Now the parties will use this new model uploaded on the blockchain and start the learning again.

4.3 Blockchain based federated learning framework design

Blockchain based federated learning framework design achieves to train the machine learning model in a decentralized and secure manner.

- The participants have a set of data with them. In every iteration of the model training process, every participant would simultaneously train the global model by using their data, after training it will ask other participants to send their local updated parameters and would also send its updated parameters to other them. This means we now have the model parameter aggregation done at each participants end and will not require a central entity for this operation. To achieve this, we can employ the gossip protocol of blockchain.

- Now after receiving and sending the model parameters, the participants will aggregate the received models and update the global model. We assume that the participants aggregate the model learning parameters honestly.

Lastly, the participants would store and record the aggregated model and would publish the blocks using blockchain consensus protocols, for example, proof of work, proof of stake etc. If we take the example of PoW, if a participant is successful in solving the problem, he will then broadcast the solution to the problem and the aggregated result to other participants and they would verify it. If they verify it, then they would add this block.

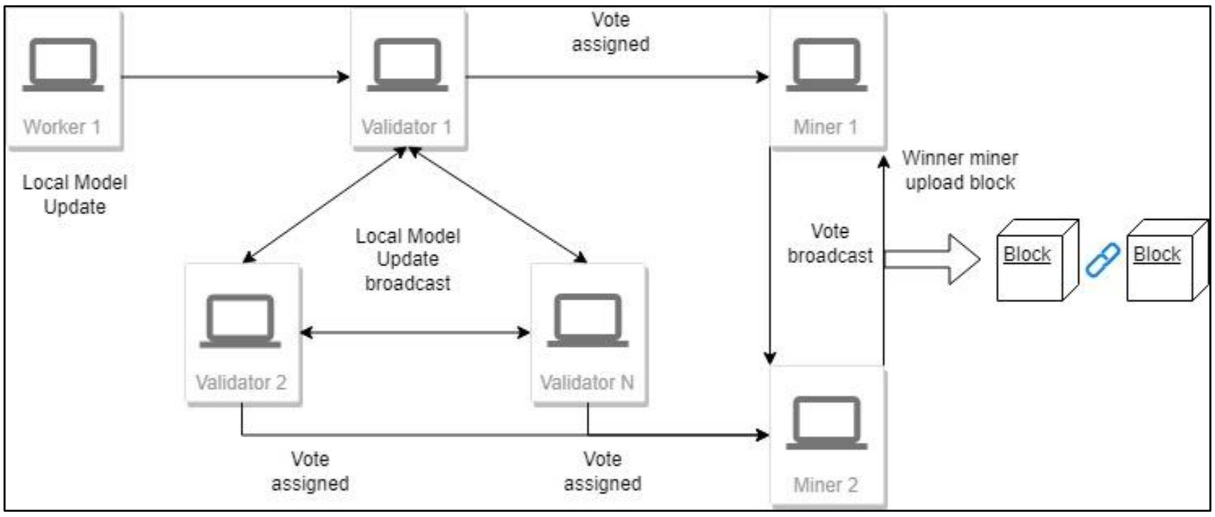


Figure 4.1: Proposed Architectural Diagram

4.4 Running the experiment

The code is written in python language. To run the experiment, we need to install Pytorch, pycryptodome and matplotlib, then create a new anaconda environment and activate that environment. Running the python file requires some arguments to be passed with. These are as follow.

Federated Learning Arguments

- We need to provide the total number of participants using argument “-nd”.
- We need to provide the local training batch size using argument “-B”.
- We need to provide the model’s name using argument “-mn”.
- We need to provide the learning rate using argument “-lr”.
- We need to provide the optimizer using argument “-op”.
- We need to provide the way to allocate data using argument “-iid”.
- We need to provide the maximum number of communication rounds using argument “-max_ncomm”.
- We need to provide the number of malicious participants using argument “-nm”.
- We need to provide the noise induced by malicious participants using argument “-nv”.
- We need to provide the local number of epochs using argument “-le”.

Blockchain arguments

- We need to provide the type of consensus we want to use using argument “-pow”. If the value passed to this argument is 0, this will mean we want to use proof of stake as the consensus mechanism, whereas if we set this argument as 1 or 2, this means that we want to use proof of work as consensus mechanism.

Participant’s arguments

- We need to distribute roles to participants using argument “-ha”, order of which will be as “worker, validator, miner”.

Command

For the purpose of explanation, we consider following command.

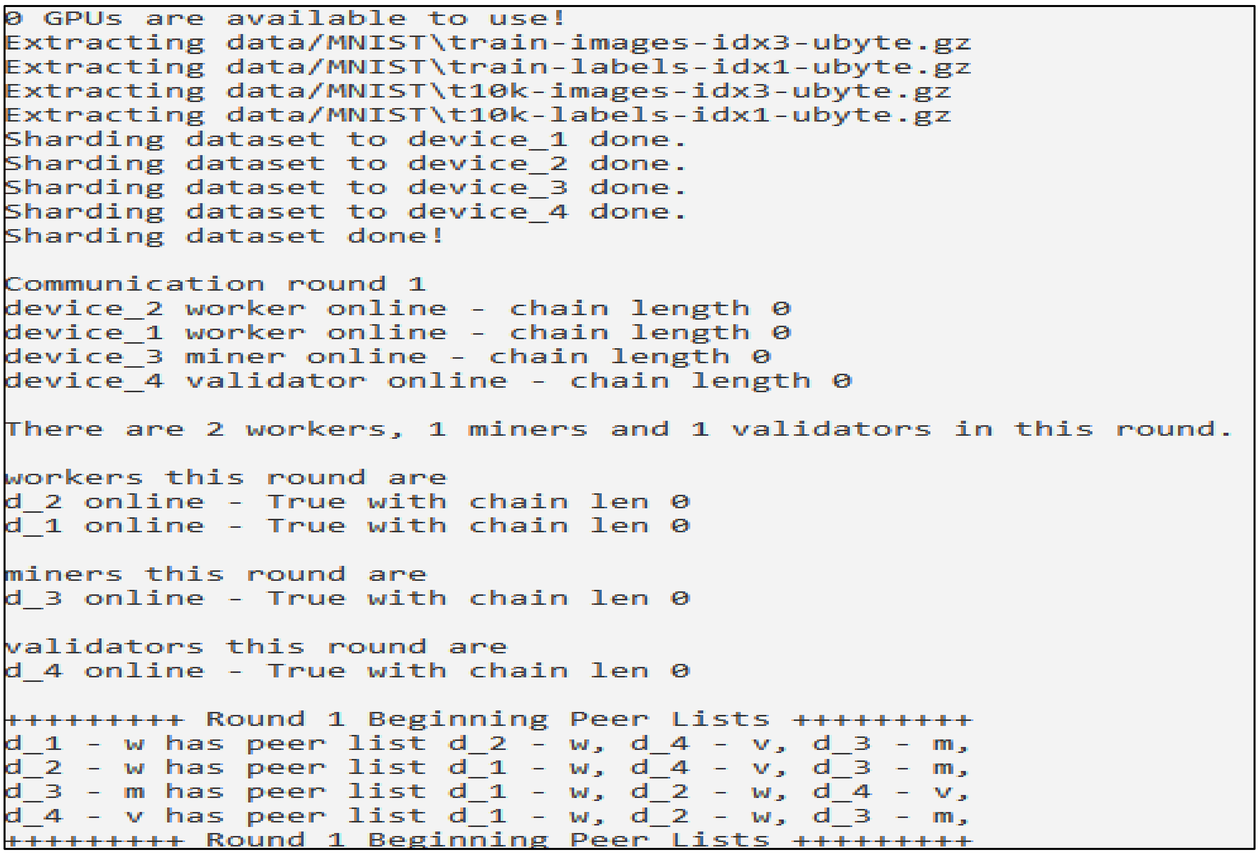
- python main.py -nd 4 -max_ncomm 2 -ha 2,1,1 -aio 1 -pow 2 -ko 6 -nm 3 -vh 0.08 -cs 0 -B 10 -mn mnist_cnn -iid 0 -lr 0.1 -dtx 1 -le 1

Here our code main.py is ran using python by passing arguments as considering 4 participants in total, among them 2 are assigned the role of workers, 1 as validator and 1 as miner and ran the communication rounds for 2 rounds, the consensus mechanism used was proof of work with a difficulty of 2. The local epoch was set to 1, the batch size was 10. The cnn model was trained on iid mnist dataset.

Example Transaction Steps

Step 0: Start of Communication round

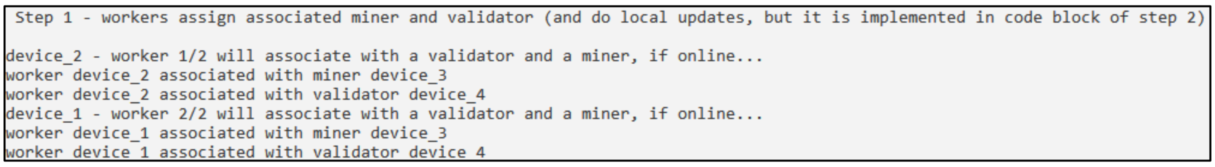
In this step the MNIST dataset is downloaded and sent to all the participants, along with this role are randomly assigned to participants as worker, validator and miner. Every participant gets register with every other participant and in this way, it makes its own peer list. Here chain length is zero, this means that it will be the genesis block.



```
0 GPUs are available to use!
Extracting data/MNIST\train-images-idx3-ubyte.gz
Extracting data/MNIST\train-labels-idx1-ubyte.gz
Extracting data/MNIST\t10k-images-idx3-ubyte.gz
Extracting data/MNIST\t10k-labels-idx1-ubyte.gz
Sharding dataset to device_1 done.
Sharding dataset to device_2 done.
Sharding dataset to device_3 done.
Sharding dataset to device_4 done.
Sharding dataset done!

Communication round 1
device_2 worker online - chain length 0
device_1 worker online - chain length 0
device_3 miner online - chain length 0
device_4 validator online - chain length 0

There are 2 workers, 1 miners and 1 validators in this round.

workers this round are
d_2 online - True with chain len 0
d_1 online - True with chain len 0

miners this round are
d_3 online - True with chain len 0

validators this round are
d_4 online - True with chain len 0

+++++++++ Round 1 Beginning Peer Lists +++++++++
d_1 - w has peer list d_2 - w, d_4 - v, d_3 - m,
d_2 - w has peer list d_1 - w, d_4 - v, d_3 - m,
d_3 - m has peer list d_1 - w, d_2 - w, d_4 - v,
d_4 - v has peer list d_1 - w, d_2 - w, d_3 - m,
+++++++++ Round 1 Beginning Peer Lists +++++++++
```

Figure 4.2: Communication round 1 and participants role assignment

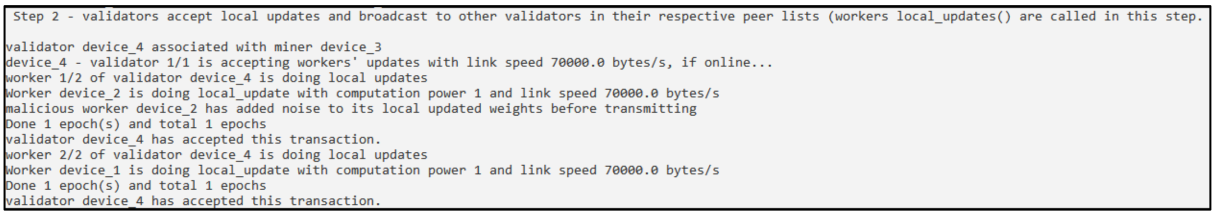
Step 1 - workers assign associated miner and validator



```
 Step 1 - workers assign associated miner and validator (and do local updates, but it is implemented in code block of step 2)

device_2 - worker 1/2 will associate with a validator and a miner, if online...
worker device_2 associated with miner device_3
worker device_2 associated with validator device_4
device_1 - worker 2/2 will associate with a validator and a miner, if online...
worker device_1 associated with miner device_3
worker device_1 associated with validator device_4
```

Figure 4.3: workers get assigned to validator and miner

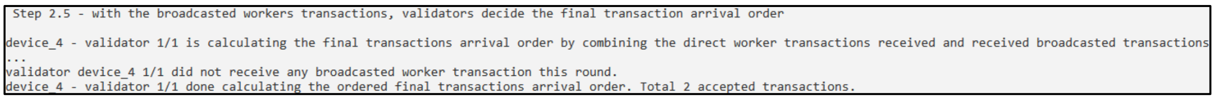
Step 2 - validators accept local updates and broadcast to other validators in their respective peer lists (workers local updates) are called in this step



```
 Step 2 - validators accept local updates and broadcast to other validators in their respective peer lists (workers local_updates() are called in this step.

validator device_4 associated with miner device_3
device_4 - validator 1/1 is accepting workers' updates with link speed 70000.0 bytes/s, if online...
worker 1/2 of validator device_4 is doing local updates
Worker device_2 is doing local_update with computation power 1 and link speed 70000.0 bytes/s
malicious worker device_2 has added noise to its local updated weights before transmitting
Done 1 epoch(s) and total 1 epochs
validator device_4 has accepted this transaction.
worker 2/2 of validator device_4 is doing local updates
Worker device_1 is doing local_update with computation power 1 and link speed 70000.0 bytes/s
Done 1 epoch(s) and total 1 epochs
validator device_4 has accepted this transaction.
```

Figure 4.4: local model updates are broadcasted to other validators

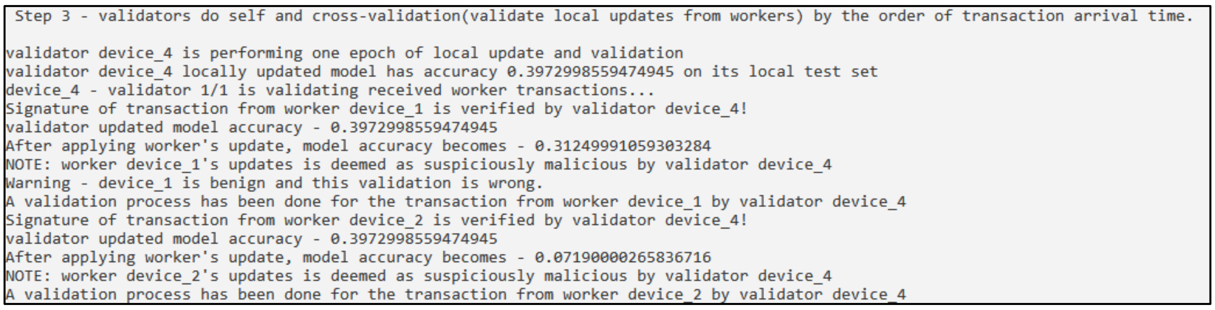
Step 2.5 - with the broadcasted workers transactions, validators decide the final transaction arrival order



```
 Step 2.5 - with the broadcasted workers transactions, validators decide the final transaction arrival order

device_4 - validator 1/1 is calculating the final transactions arrival order by combining the direct worker transactions received and received broadcasted transactions
...
validator device_4 1/1 did not receive any broadcasted worker transaction this round.
device_4 - validator 1/1 done calculating the ordered final transactions arrival order. Total 2 accepted transactions.
```

Figure 2.5: Transaction arrival order

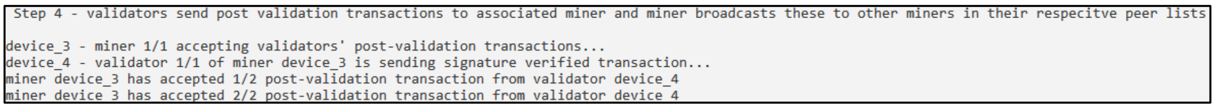
Step 3 - validators do self and cross-validation (validate local updates from workers) by the order of transaction arrival time



```
 Step 3 - validators do self and cross-validation(validate local updates from workers) by the order of transaction arrival time.

validator device_4 is performing one epoch of local update and validation
validator device_4 locally updated model has accuracy 0.3972998559474945 on its local test set
device_4 - validator 1/1 is validating received worker transactions...
Signature of transaction from worker device_1 is verified by validator device_4!
validator updated model accuracy - 0.3972998559474945
After applying worker's update, model accuracy becomes - 0.31249991059303284
NOTE: worker device_1's updates is deemed as suspiciously malicious by validator device_4
Warning - device_1 is benign and this validation is wrong.
A validation process has been done for the transaction from worker device_1 by validator device_4
Signature of transaction from worker device_2 is verified by validator device_4!
validator updated model accuracy - 0.3972998559474945
After applying worker's update, model accuracy becomes - 0.07190000265836716
NOTE: worker device_2's updates is deemed as suspiciously malicious by validator device_4
A validation process has been done for the transaction from worker device_2 by validator device_4
```

Figure 4.6: Validators performing self and cross validation

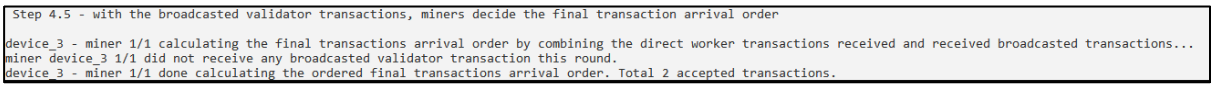
Step 4 - validators send post validation transactions to associated miner and miner broadcasts these to other miners in their respective peer lists



```
 Step 4 - validators send post validation transactions to associated miner and miner broadcasts these to other miners in their respecitve peer lists

device_3 - miner 1/1 accepting validators' post-validation transactions...
device_4 - validator 1/1 of miner device_3 is sending signature verified transaction...
miner device_3 has accepted 1/2 post-validation transaction from validator device_4
miner device_3 has accepted 2/2 post-validation transaction from validator device_4
```

Figure 4.7: Validator sends transactions to associated miner

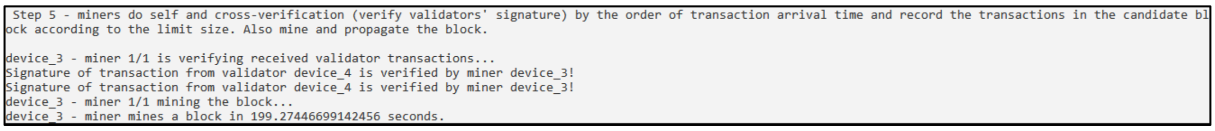
Step 4.5 - with the broadcasted validator transactions, miners decide the final transaction arrival order



```
 Step 4.5 - with the broadcasted validator transactions, miners decide the final transaction arrival order

device_3 - miner 1/1 calculating the final transactions arrival order by combining the direct worker transactions received and received broadcasted transactions...
miner device_3 1/1 did not receive any broadcasted validator transaction this round.
device_3 - miner 1/1 done calculating the ordered final transactions arrival order. Total 2 accepted transactions.
```

Figure 4.8: Transaction arrival order

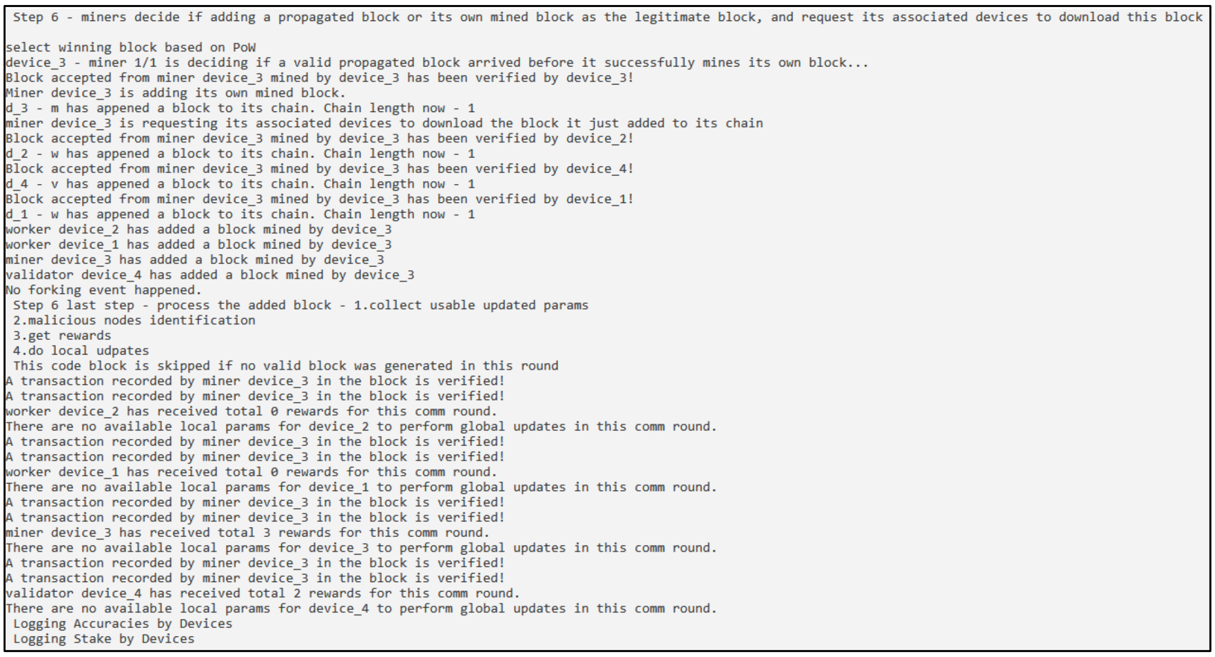
Step 5 - miners do self and cross-verification (verify validators' signature) by the order of transaction arrival time and record the transactions in the candidate block according to the limit size. Also, mine and propagate the block



```
 Step 5 - miners do self and cross-verification (verify validators' signature) by the order of transaction arrival time and record the transactions in the candidate bl
ock according to the limit size. Also mine and propagate the block.

device_3 - miner 1/1 is verifying received validator transactions...
Signature of transaction from validator device_4 is verified by miner device_3!
Signature of transaction from validator device_4 is verified by miner device_3!
device_3 - miner 1/1 mining the block...
device_3 - miner mines a block in 199.27446699142456 seconds.
```

Figure 4.9: Miner doing self and cross validation

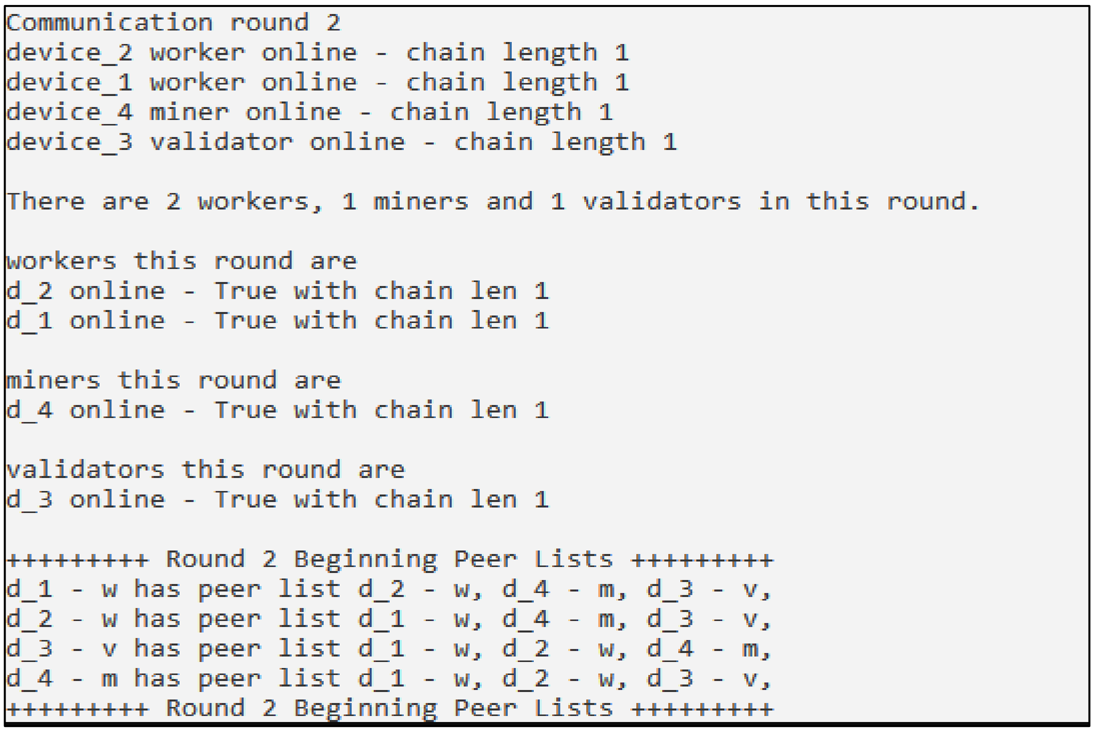
Step 6 - miners decide if adding a propagated block or its own mined block as the legitimate block, and request its associated devices to download this block



```
 Step 6 - miners decide if adding a propagated block or its own mined block as the legitimate block, and request its associated devices to download this block

select winning block based on PoW
device_3 - miner 1/1 is deciding if a valid propagated block arrived before it successfully mines its own block...
Block accepted from miner device_3 mined by device_3 has been verified by device_3!
Miner device_3 is adding its own mined block.
d_3 - m has appened a block to its chain. Chain length now - 1
miner device_3 is requesting its associated devices to download the block it just added to its chain
Block accepted from miner device_3 mined by device_3 has been verified by device_2!
d_2 - w has appened a block to its chain. Chain length now - 1
Block accepted from miner device_3 mined by device_3 has been verified by device_4!
d_4 - v has appened a block to its chain. Chain length now - 1
Block accepted from miner device_3 mined by device_3 has been verified by device_1!
d_1 - w has appened a block to its chain. Chain length now - 1
worker device_2 has added a block mined by device_3
worker device_1 has added a block mined by device_3
miner device_3 has added a block mined by device_3
validator device_4 has added a block mined by device_3
No forking event happened.
 Step 6 last step - process the added block - 1.collect usable updated params
 2.malicious nodes identification
 3.get rewards
 4.do local udpates
 This code block is skipped if no valid block was generated in this round
A transaction recorded by miner device_3 in the block is verified!
A transaction recorded by miner device_3 in the block is verified!
worker device_2 has received total 0 rewards for this comm round.
There are no available local params for device_2 to perform global updates in this comm round.
A transaction recorded by miner device_3 in the block is verified!
A transaction recorded by miner device_3 in the block is verified!
worker device_1 has received total 0 rewards for this comm round.
There are no available local params for device_1 to perform global updates in this comm round.
A transaction recorded by miner device_3 in the block is verified!
A transaction recorded by miner device_3 in the block is verified!
miner device_3 has received total 3 rewards for this comm round.
There are no available local params for device_3 to perform global updates in this comm round.
A transaction recorded by miner device_3 in the block is verified!
A transaction recorded by miner device_3 in the block is verified!
validator device_4 has received total 2 rewards for this comm round.
There are no available local params for device_4 to perform global updates in this comm round.
 Logging Accuracies by Devices
 Logging Stake by Devices
```

Figure 4.10: block is propagated and downloaded by all participants

Now after the block is propagated and downloaded by all participants, the second round will start. Again, the participants will be randomly assigned roles and one difference that we can see here is that now the chain length has changed from 0 to 1. This means the block is appended.



```
Communication round 2
device_2 worker online - chain length 1
device_1 worker online - chain length 1
device_4 miner online - chain length 1
device_3 validator online - chain length 1

There are 2 workers, 1 miners and 1 validators in this round.

workers this round are
d_2 online - True with chain len 1
d_1 online - True with chain len 1

miners this round are
d_4 online - True with chain len 1

validators this round are
d_3 online - True with chain len 1

+++++++++ Round 2 Beginning Peer Lists +++++++++
d_1 - w has peer list d_2 - w, d_4 - m, d_3 - v,
d_2 - w has peer list d_1 - w, d_4 - m, d_3 - v,
d_3 - v has peer list d_1 - w, d_2 - w, d_4 - m,
d_4 - m has peer list d_1 - w, d_2 - w, d_3 - v,
+++++++++ Round 2 Beginning Peer Lists +++++++++
```

Figure 4.11: Second communication round begins

Chapter 5

Experimental Setup

5.1 Chapter Outline

Discussion on which consensus mechanism could be best for federated learning under different conditions. Use cases of proposed consensus mechanism for federated learning framework and then propose which mechanism is good.

In FL settings, a server performs the central operations of update aggregation, client selection, and global model maintenance. The server needs to collect updates from numerous clients for aggregation operation, and it also needs to broadcast new global models to these clients, which puts a high demand on network bandwidth. Also, cloud-based servers are affected by the stability of cloud service providers [44]. A centralized server can skew the global model by favoring some clients. Malicious central servers can poison the model and even collect clients' privacy from updates. Therefore, the stability, fairness, and security of the central server are crucial to FL.

However, the current federated learning system relies on a central coordinator, so there are still problems of single point failure and trust. Blockchain is a decentralized, traceable, tamper-proof distributed ledger that can provide data security and data validation for federated learning [45], [46], ensuring data security and consistency of model parameters among untrusted participants [47]. Despite blockchain is an effective solution to replace the attack-prone central coordinator in the federated learning system, the combination of federated learning and blockchain may bring potential security and privacy attacks. First of all, all local updates are recorded as plain texts in blocks. Curious participants or servers can infer sensitive information of a target participant through observation of the local updates submitted by him. Secondly, there is no guarantee that there are no malicious participants alone or in collusion among multi-party entities in a distributed network. Malicious participants may purposefully modify the label of local data and intentionally send adversarial local updates to deteriorate the accuracy of the shared model. The system needs to be able to withstand some security attacks, such as collusion attack, sybil attack, poisoning attack, and inference attack [48].

It is reasonable to assume that the clients in FL might be malicious. Therefore, the local updates from all clients should be recorded under blockchain-based FL settings.

However, the works did not focus on the consensus protocols regarding poisoning attacks, which left room for performance degradation and vulnerability if there exist malicious and/or compromised nodes in the system uploading noisy local updates for global model construction.

5.1.1 Scenario: Malicious participants becoming miner

In blockchain based federated learning system we have various participants. Among them are the workers, validators and miners. Miners who are there to collect the validated local model updates from every other participant in the system and then after collecting the local model updates it performs the aggregation of the validators assigned votes. Appending the local model updates and the votes of the workers on the blockchain is also done by miners. If we assume miners to act maliciously such that it attempts to insert corrupted and wrong updates to harm the process of global model creation. This issue is addressed such that the system does not opt for the block mined by a corrupted participant.

Solution: -

1. Local model updates are validated through a voting-based validation mechanism.
2. Honest participant's updates are rewarded through a proof of stake consensus mechanism and then those honest participants are given the chance to generate blocks for recording the updates.

We consider three roles for the participants.

a. Workers (those who posse local dataset and are doing machine learning).
b. Validators (those who are responsible for the validation of the updates from the trainers and then assigning them votes accordingly).
c. Miners (those who will append the updates with the voting given by the validators and then add them to the block).

Steps: -

1. Validating the model updates:
   After the worker is done with the machine learning process, it will send the updates to the validators, who will try to validate the updates from the trainers by using their local data and then forward this to the miners. Now we can assume that if the malicious worker sends wrong models, then it must affect the performance (this can be either way,

performance level can be increased or decreased). Ideally malicious workers would wish to increase the performance level quickly. This can be used as an indicator to observe the performance of the malicious workers.

2. Proof of Stake consensus:
   Stake means a reward, proof of stake consensus mechanism in our setting can be beneficial to limit the chances of malicious workers/ validators/ miners assuming the role of miner in each round. The idea behind proof of stake in blockchain based federated learning setup is that the miner from the pool of miners, which will do the most contribution will get the chance to publish the block (the block will contain the updates of the local model and the votes for that update). Now we distribute the rewards to the miners, and this will either increase or decrease the overall stake of that participant in the network. This stake is also documented into the blockchain.

3. The validation process decreases the likelihood of malicious participants to get frequent rewards because the stake of the malicious participants would decrease.

4. Compared with the poof of work consensus mechanism, if we employ proof of stake as the consensus mechanism it would be less computationally expensive and communication efficient.

Experimental Setup: -

Firstly, we will discuss how the proof of stake based blockchain enabled federated learning system operates. We consider participants which will do the learning work in iterative rounds. The participants are further categorized into three participants (workers, validators and miners). For the verification of the blocks, we assumed participant devices to have respective ids.

i. Worker: -

In any round the worker "**w**" will generate a local model "**L**" update by doing learning over the previous round's global model "**G**". We observe that in Vanilla federated learning rounds "**R**" the global model was directly calculated from the local model updates. Whereas, in proof of stake we introduce the concept of validation and hence the local model updates will firstly go to the validators for validation purposes and then the global model is calculated. After the learning process the worker will sign the transaction “**tx**” using his private key and then send it for validation.

ii. Validator: -

The transactions are then passed to each validator from each worker. Votes to the local updates are given by the validators and if the local model votes are +ve, it will result in workers gaining the rewards “**r**”. Similarly, validators will also be rewarded based on the verification they have performed over the local model updates from workers. The transaction received from the worker contains the local model updates, this local model update will be given a vote from the validator as +ve or -ve. Voting is done by the validator such that the validator will use its training data to perform the local model update. Now this transaction will be privately signed by the validator and sent to the miner.

iii. Miner: -

The transactions received from the validator to the miners. Now the miner will have the local model update done by the worker, the voting of that local model update done by the validator. Similarly, like workers and validators the miners will also get the reward for the work it will do. After verification of the signature the miner would take the local model updates from the transaction it received from the validator. Now the miner will aggregate all the votes given to the local model update from validators and will then be appended into a block. Miner would now start the 0-difficulty mining process for the block it created by hashing it.

For the purpose of experiment, we will assume to train on MNIST dataset. We assume the blockchain based federated learning system architecture consist of following participants.

1. Worker = 2
2. Validators = 3
3. Miners = 2

In round **R1**, the two workers **w1 and w2** will perform the learning task and will create two Local models **L1 and L2**. These updates will be forwarded to the three validators **v1, v2 and v3**.

Here we will consider two possible outcomes.

a. Validator **v1** and **v3** voted positively against the local model **L1** of worker **w1**. Validator **v2** votes negatively. The two miners **m1 and m2** will aggregate these votes from validator against the two local models. It is evident that the result of the aggregation of **w1**'s local update votes calculated by the two miners will be the same (2 +ve and 1 -ve).

b. All the three validators gave -ve votes to the local model update of worker w2.
i.e.,

Vote of v1 for w1 = +Ve

Vote of v2 for w1 = +Ve

Vote of v3 for w1 = -Ve

Vote of v1 for w2 = -Ve

Vote of v2 for w2 = -Ve

Vote of v3 for w2 = -Ve

c. The two miners **m1 and** m2, will now create their respected blocks **b1 and b2** and will append the two worker updates in them.
i.e.,

Block of miner ***m1*** ***= L1 + L2***

Block of miner ***m2*** ***= L1 + L2***

iv. Mining process:

In mining process one miner would mine the block and then the block will be sent to rest of the miners. This process will be carried out by all other miners. The block that contains the greatest stake will be selected as the rightful block. Now in this block we will have the transactions (rewards, votes and model update).

v. Participants task after appending block:

Following two tasks will be performed by the participants after blockchain is updated.

a. Global model update:
The participants will update the global model by using the local model updates with the greatest +ve votes.

b. Stake update:

Participants would then update the stakes of its peers.

Let us continue with our example where we supposed that we have two miners, **m1 and m2** and **m1** had lesser stake than miner **m2.** This will mean that the block will be mined by the miner having greater stake. We consider the same previous transactions for the mined block in current example.

The transactions from our example were

Worker 1 = 2 +ve and 1 -ve votes for worker w1 updates, and

Worker 2 = 0 +ve and 3 -ves for w2 updates.

Now that miner m2 has mined the block, this block will then be propagated to rest of the participants.

a. Here, this is evident that worker w1 local model updates will be used to create the next global model as the number of +ve votes are greater than the -ve votes. After global model calculation, participants will update the stake values. This stake is helpful in determining the trustful block from malicious block, because a higher stake will indicate a benign block.
b. On the other hand, in our example if we assume the case in which after mining the block the local model updates of worker w2 is considered instead of w1, then the worker w2 will be considered as a malicious worker. Now if these workers updates are considered over few rounds, the participants would discard the coming updates from that worker. This is how proof of stake can be used to differentiate between the benign and malicious transactions from workers.

Through the validation process we can distinguish between the benign and malicious workers. By malicious workers we refer to the possible situation where workers add extra noise in their updates to corrupt the global model update process. The validation process is such that the validator will take the previous rounds global model's accuracy and will try to match it with the current round's local updates accuracy.

Accuracy of workers local model = Accuracy of previous round Global model

Now in a situation where malicious workers had corrupted the updates the accuracy will likely go down when matched with pervious global models' accuracy. Otherwise, the accuracy will go up if the updates are not corrupted.

vi. Validator trusting workers local model updates:

Another concern that arises is of validators believing the workers with their local updates. As the validators are unknown to the training and testing dataset of the workers and the validators only receives the local updates from the workers and must trust the workers for calculating the local model accuracy against the previous rounds global model accuracy.

Validator receives from workers = {Accuracy of workers local model, Accuracy of previous round Global model}

One possible solution to this issue is that the updates received from workers by the validators must be validated by using validators own test data and calculate the accuracy of both the received local model and the previous global model. We assumed this possibility on the basis that the workers and validators have the same dataset. This results in.

Accuracy of validator calculated for worker's local model = Accuracy of worker calculated for its own local model.

Accuracy of validator calculated for previous Global model = Accuracy of worker calculated for previous Global model

vii. Threshold (vh) calculation:

To distinguish between the benign and malicious workers, there must be a threshold level which will guide in marking the participants. We can have following two scenarios.

a. *Accuracy of validator calculated for previous round's global model - Accuracy of validator calculated for worker's local model update > Threshold value (vh).* Validator would then treat the local model update as well as the worker as malicious participants and will assign a -ve vote.
b. *Accuracy of validator calculated for previous round's global model - Accuracy of validator calculated for worker's local model update < Threshold value (vh).*
Validator would then treat the local model update as well as the worker as benign and assign a +ve vote.

Experiment with fixed threshold values: -

We consider 20 participants (12 workers, 5 validators and 3 miners) and divided them such that we have 17 good participants and rest are 3 bad participants. The experiment was carried out using the MNIST dataset. To demonstrate a malicious worker sending corrupted updates, we tried to add noise in the local model update. Noise of +1 is added to the local model update and then sent to the validator.

To observe if proof of stake based blockchain enabled federated learning setting can distinguish between the malicious and good participants, we assume some threshold values (vh) to be as 0.01, 0.1, 0.2. it was observed that there was no distinguishable difference between the threshold values the accuracy of validator for global model and local update.

To compare with the Vanilla federated learning, the experiment assumed 20 participants, for 100 rounds every participant performed 5 local epochs.

a. Accuracy of the worker calculated for the global model is represented with orange line.
b. Horizontal axis depicts accuracy of worker calculated for its own local model update.

Analysis:

It was seen that accuracy between workers local model update and previous round's global model had a great difference. In each round every worker had to perform local training of 5 epochs. Good participants were seen to have steep lower accuracy. This trend was observed for remaining 90 rounds.

Amendments:

We assume to run the first epoch in non-malicious way. This way we can eliminate the accuracy of validator for previous round's global model. Now after one epoch the accuracy of validator for 1st round of validators local model update. This accuracy is assumed to be equal to the accuracy of the worker for 1st round of workers local model update. Now this accuracy of validator for 1st round of validators local model update will be used to compute the variation of validation accuracy.

Validation accuracy = Accuracy of validator calculated for 1st round of validator's local model update – Accuracy of validator calculated for nth round of workers local model update

Now to observe that the local model update is corrupted or not, this validation accuracy will then be evaluated against the threshold.

Experiment with validation accuracy:

We assume same experimental setup, we assume to have three malicious participants and run it for 30 rounds.

Here we observe red colored dots showing validation accuracy scores by the malicious participants and green colored dots showing accuracy of good participants. It was seen that validation accuracy greater than 0.08 had most red dots and this could be used as a threshold to determine that above 0.08 the model updates would be corrupted, and we can mark that worker as malicious.

Results and Discussion: -

Through PoS we aim to safeguard the trustworthiness of the local model updates and store them safely on blockchain which will then be used to make the global model. Through proof of stake consensus mechanism, we go for selecting the miners in the system. We observed that how Proof of stake rewarding system rewards the worker who contributed effectively. The more the stake, it means the larger that participant has contributed to the system. Now the most important part is selecting the right block for updating the global model. Still, we observe that greater the stake means greater work was done by that participant and greater the probability of selecting its block.

For proof of stake-based system and for Vanilla FL, Federated Averaging method was employed, and the dataset used was MNIST to train a CNN network. In one single round the participants will ran for 5 epochs with a learning rate of 0.01 and the batch size was 10.

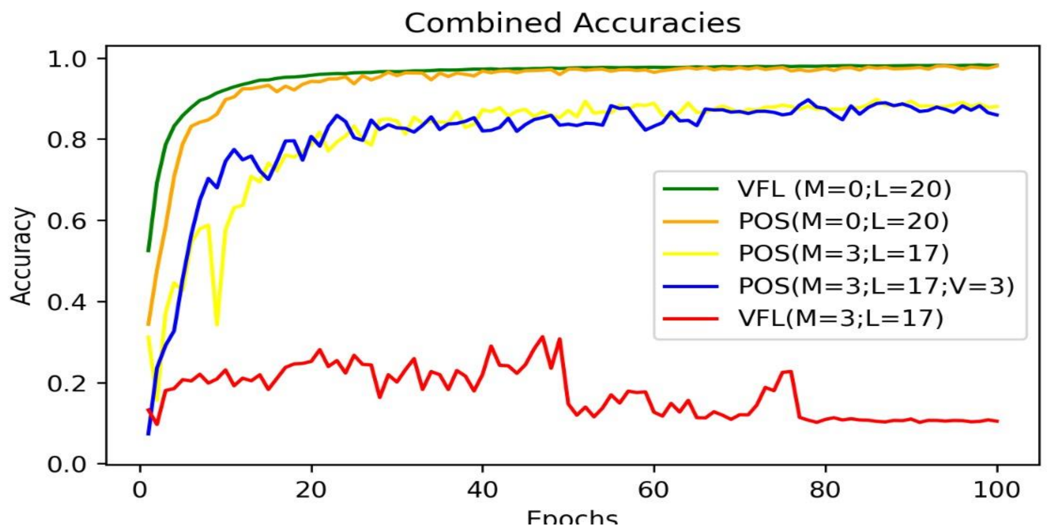


Figure 5.1: Combined; Vanilla Federated Learning with 0 malicious participants V/s PoS with 0 malicious Vs PoS with 3 malicious Vs PoS with 3 Malicious Validators Vs Vanilla FL with 3 malicious

Legend:

a. Vanilla Federated learning when all the participants in the system are good. – *orange line.*
b. Vanilla Federated learning when 17 participants are good but 3 are malicious participants. – *red line.*
c. PoS -Blockchain based Federated learning when all 20 participants are good. – *green line.*
d. PoS - Blockchain based Federated learning with 17 participants are good, 3 participants are malicious. – *blue line.*
e. PoW - Blockchain based Federated learning 17 participants are good, 3 participants are malicious.

   - No malicious participant

   The experiment was running with 20 participants (12 workers, 5 validators and 3 miners) for 100 rounds and 3 local epochs. It is observed that all the local model updates will be used to calculate the global mode if threshold value is 1 and the validators vote +ve for local model update. When there are no malicious participants, it is seen that both Vanilla Federated learning and PoS based Blockchain enabled Federated learning performed almost the same.

   - With Malicious participants

   It was seen that even when the malicious participant assumed the roles of 'm' and 'v' there was no malicious activity. With 3 malicious participants it was seen that it does not utilize the corrupted local models from those three participants and the accuracy reaches to that observed with Vanilla FL (orange and green).

   - Malicious participant assumed the role of validator

   It was seen that under such situation the participant will turn over the votes. Through brown line it is seen that if there are few malicious participants it will not affect the process.

It was observed that making 15% of the participants to play a malicious role, the system achieves an accuracy level of 87%.

We further compared the Proof of Stake based blockchain enabled federated learning system with the traditional Vanilla federated learning system. It is concluded that the Vanilla Federated learning system did not employ Proof of Stake as consensus mechanism and there was no validation scheme introduced.

A. Vanilla federated learning with 0 malicious participants V/s Vanilla federated learning with 3 malicious participants.

To begin with we have considered Vanilla federated learning as our benchmark. As seen through the Figure 1, with 20 participants we ran the system for 100 rounds. Similarly, we ran the system for 100 rounds with 3 malicious participants.

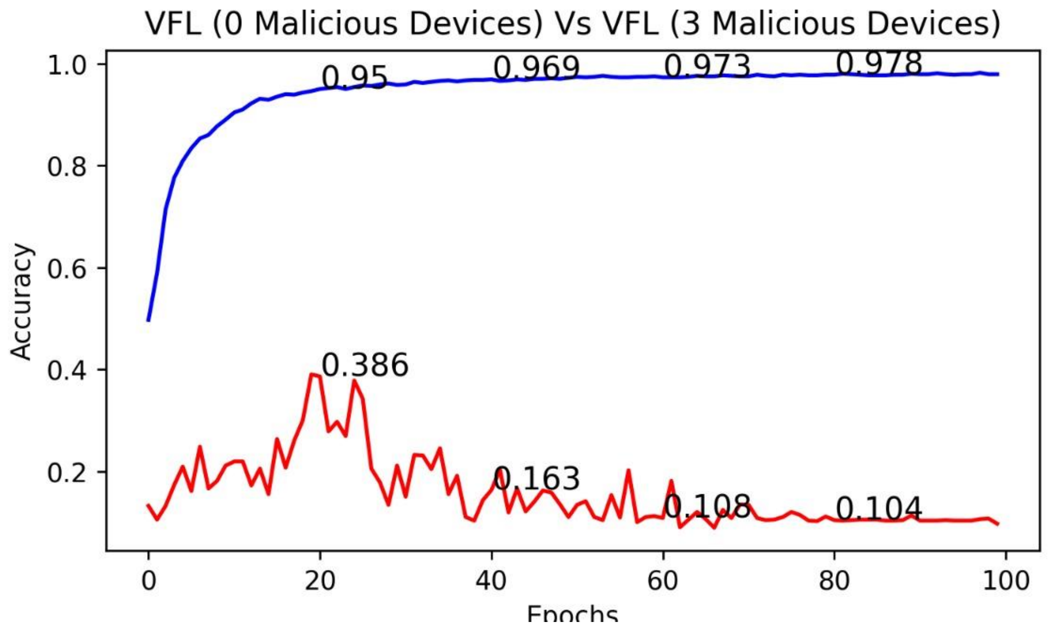


Figure 3: Comparison of Simple Vanilla Federated Learning with 0 malicious participants against 3 malicious participants.

Analysis:

• As seen through Figure 1, the accuracy at the end of 100 training rounds, it has reached to 97.

• It is seen that the accuracy of training after 100 rounds has drop considerably.

• This is because in Vanilla federated learning, there is no method of validating the local model updates from the participants. The central aggregator would receive the updates from participants and would aggregate them and send the global model back to participants for next round.

B. **Vanilla federated learning with 0 malicious V/s Proof of Stake based BFL with 0 malicious participants.**

Next, we considered comparing the performance of Vanilla federated learning with blockchain based federated learning. In both cases we ran the system for 100 rounds and there were no malicious participants.

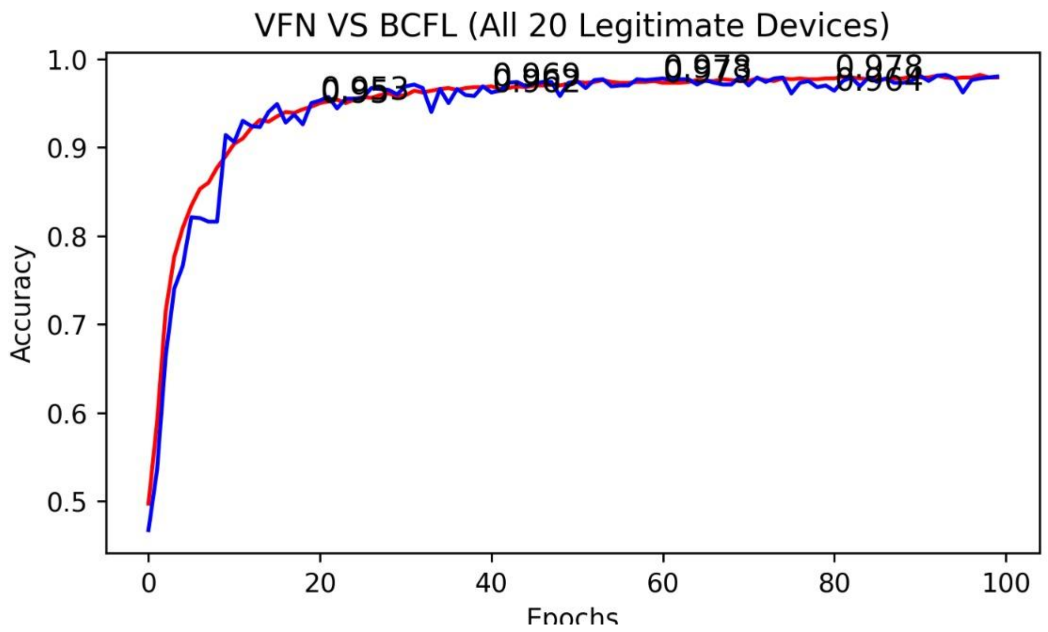


Figure 4: Vanilla federated learning with 0 malicious participants V/s blockchain based FL with 0 malicious participants

Analysis:

• It was observed that with no malicious participants involved in the training rounds, both vanilla federated learning as well as blockchain based federated learning achieves almost the same accuracy. The only drawback is that in blockchain based federated learning we face a huge delay in training rounds.

Proof-of-Stake vs Proof-of-Work

Experiment setup:

We do have miners in proof of work consensus mechanism, this gives us the leverage to compare the two consensus mechanisms from the point of selecting malicious participants in proof of work. In proof of work mining we have a concept of nonce for the block. After the miner is able to guess the correct nonce it will be able to send this block to other miners and append the block to the blockchain. Once the trustworthy block is received the mining stops.

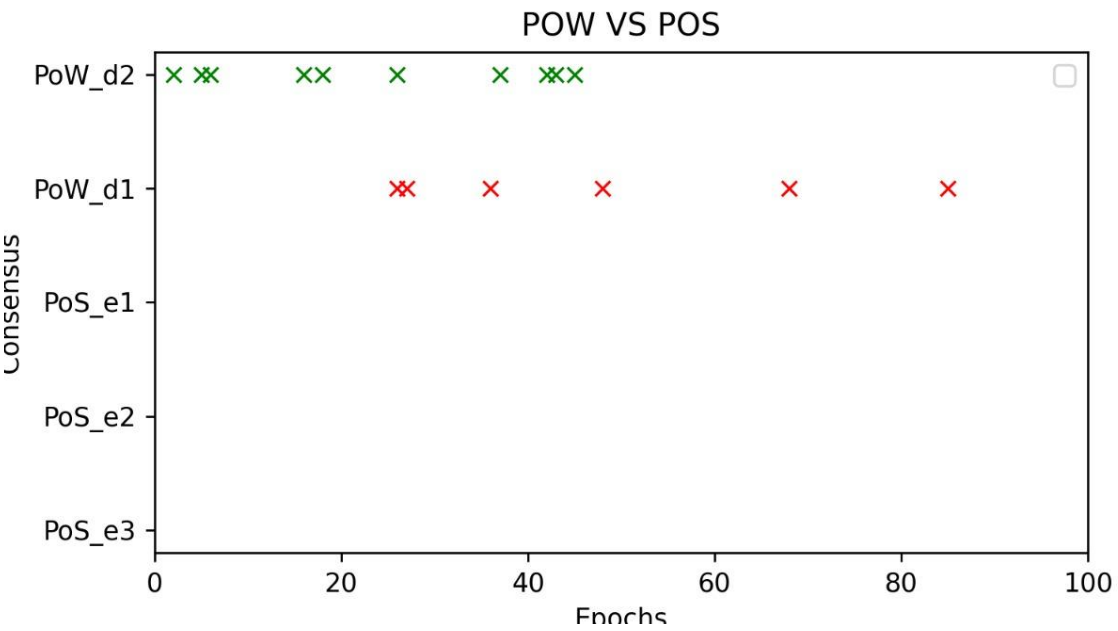


Figure 5: Proof of Work Vs Proof of Stake

Case 2: Byzantine Fault-Tolerance (participants try to become validator)

Update the scenario with this: -

One of the most common and challenging attacks in decentralized systems is Byzantine attack, where an attacker follows the system protocol but propagates arbitrary malicious information to benign system participants, aiming to degrade the system performance and further mislead or control the system output. The main processes in the proposed system include model collection (i.e., collecting local models from other parties) and model update (i.e., sending calibration messages to parties if needed).

Scenario:

A decentralized system such as blockchain based federated learning system is most likely prone towards byzantine attacks. Here, we consider a possible case scenario in which we have participants that are not contributing with useful updates to the global model, and we can call them as lazy participants. Others are malicious trainers.

Consensus mechanism: practical byzantine fault tolerance

Solution: Since the byzantine problem is inspired by real byzantine general problem, where the generals come to a consensus of successfully attacking the fort at the same time and if there is a traitor general the attack won't be successful, and his reputation will be lost. Similarly, when applying the concept to our threat model we can say that to provide a solution to the byzantine problem we assign participants according to their reputation i.e., with a trust score. Now for the trainer participants this trust score can be made as a weight factor in the federated learning process. When we have made trust score as the weight parameter it will have following effects.

1. Validators will be able to control the trainer's updates.
2. If one trainer reputation (i.e., trust score) is changed it will influence the reputation of other trainers as well.

When the trainer's updates are validated by the validators, it will change the trust scores of the trainers. The process of changing the trust score is based on the fact that whether there is an increase or decrease in the performance. Consistency of data among the validators is maintained by using practical byzantine fault tolerance consensus mechanism, it is based on the voting or elections system.

Byzantine fault tolerance BFL framework: -

The system architecture can be thought of involving multiple participants, such that there are multiple trainers that are not directly connected to each other, whereas on the other hand we have the validators that are fully connected with each other.

Experimental Setup: -

We will take an example where we will train a CNN over MNIST dataset and compare the performance with centralized performance. We consider that for centralized system we will use the whole MNIST dataset where for decentralized system we consider trainers to keep training data set to be 30% of all the MNIST data set. While there will be 7 trainers, 3 validators and a reward – penalty policy.

We want to find out that which trainer – validator combination will result into best performance. Along with this other parameter that we want to measure are the accuracy, speed of convergence, trustworthiness. Let us suppose we have K participants (trainers and validators) and in each iteration we try a different combination of trainers and validators. Here we consider two case scenarios.

1. **More Trainers – Less Validators: -**

   It is evident that if we have good amount of data and we also increase the number of trainers we will have good performance. But on the other hand, what will happen is we will be having lesser validator participants, and if the number of validators is only one, this will have two meaning. First is that it will defeat the whole purpose of removing the dependency of centralized entity in the network. Secondly, if there is only one validator, we must trust on that validator.

2. **Less Trainers – More Validators: -**

   Having more validators in the system would increase the trustworthiness, but it will also increase the computational power required by validators to come to a consensus, because let us suppose one trainer will send its parameters to the validator and after validation it would send it to all other validators for reaching consensus.

Algorithm outline for Blockchain based Federated Learning

Algorithm 1: An outline of the Blockchain based Federated Learning Architecture

```
function MKDIR (log_files_folder_path)
        // -mkdir: create log files folder
        return ()
end function
function ROLES_REQUIRMENT (hard_assign)
        // -ha= (workers, validators, miners)
        return ()
end function
function MODEL_NAME (mnist_cnn)
        return true or false
end function
function LOSS_FUNC ()
        return true or false
end function
function MINING_CONSESNUS (PoW, PoS)
        return PoW or PoS
end function
```

MNIST database of handwritten digits

Chapter 6

Conclusion and Future Directions

In this research work, we investigated the shortcoming in current federated learning setting and selected blockchain as a potential candidate to overcome those shortcomings. For this purpose, we began our research with the focus on why there is a need to integrate the two technologies. We then illustrated the network architecture which achieves this integration of blockchain with federated learning. After this a detailed study on the security aspects of blockchain consensus mechanism was carried out, which when applied to the above blockchain based federated learning system can help better perform analysis. After this we carried out the experiments by considering malicious activities in the form of malicious updates send from workers to validators to corrupt the model accuracy. Our results suggested that when we applied proof of work as the consensus mechanism we observed events where a malicious participant was able to be selected as a miner and corrupt the model accuracy, this is because in proof of work consensus we validate the results by solving the mathematical problem and participants with greater computation power can perform greater computation and increase chances of publishing the block, whereas on the other hand in proof of stake, as the stake of malicious participants were not positively increasing, it resulted in the fact that no malicious participant was selected as a miner.

In future work, we plan to investigate other consensus mechanism and the possibility of using smart contract to authenticate and validate the results from participants. A range of attacks including denial of service, sybil attack, forking possibility, presence of malicious aggregator can be used. Moreover, if this experiment is conducted using actual representation of the environment where it will be used can much better propose its useability.

BIBLIOGRAPHY

[1] https://medium.com/swlh/a-simple-guide-to-understanding-blockchain-8dd09356b153

[2] Google AI Blog: Federated Learning: Collaborative Machine Learning without Centralized Training Data (googleblog.com)

[3] W. Gu, J. Li and Z. Tang, "A Survey on Consensus Mechanisms for Blockchain Technology," 2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA), 2021, pp. 46-49

[4] Wang, Q., Huang, J., Wang, S., Chen, Y., Zhang, P., & He, L. (2020). “A comparative study of blockchain consensus algorithms”. In Journal of Physics: Conference Series (Vol. 1437, No. 1, p. 012007).

[5] https://whatis.techtarget.com/definition/round-robin#:~:text=A%20round%20robin%20is%20an,the%20list%20and%20so%20on.&text=A%20round%2Drobin%20story%20is,successively%20by%20others%20in%20turn

[6] Zhang, Chen, et al. “A Survey on Federated Learning.” Knowledge-Based Systems, vol. 216, Elsevier BV, Mar. 2021, p. 106775.

[7] C. Zhang, X. Hu, Y. Xie, M. Gong, B. Yu, “A privacy-preserving multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition”, Front. Neurorob. 13 (2020) 112.

[8] M. Gong, J. Feng, Y. Xie, “Privacy-enhanced multi-party deep learning”, Neural Netw. 121 (2020) 484–496.

[9] Y. Xie, H. Wang, B. Yu, C. Zhang, “Secure collaborative few-shot learning”, Knowl. Based Syst. (2020) 106157.

[10] J.P. Albrecht, “How the GDPR will change the world”, Eur. Data Prot. Law Rev. 2 (2016) 287.

[11] M. Parasol, “The impact of China’s 2016 cyber security law on foreign technology firms, and on China’s big data and smart city dreams'”, Comput. Law Secur. Rev. 34 (1) (2018) 67–98.

[12] W. Gray, H.R. Zheng, “General principles of civil law of the people’s Republic of China”, Am. J. Comp. Law 34 (4) (1986) 715–743

[13] Kairouz, H.B. McMahan, B. Avent, et al., “Advances and open problems in federated learning”, December 2019.

[14] L. T. Phong, Y. Aono, T. Hayashi, et al., “Privacy-preserving deep learning via additively homomorphic encryption, IEEE Transactions on Information Forensics and Security”, 13(5):1333– 1345, May 2018

[15] K. Bonawitz, V. Ivanov, B. Kreuter, et al., “Practical secure aggregation for privacy-preserving machine learning”, In Proc. of the ACM SIGSAC Conference on Computer and Communications Security (CCS), pp. 1175–1191, November 2017.

[16] H. B. McMahan, D. Ramage, K. Talwar, et al., “Learning differentially private recurrent language models”, ArXiv Preprint ArXiv:1710.06963, October 2017.

[17] L. T. Phong, Y. Aono, T. Hayashi, et al., “Privacy-preserving deep learning via additively homomorphic encryption”, IEEE Transactions on Information Forensics and Security, 13(5):1333–1345, May 2018.

[18] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, “Deep models under the GAN: Information leakage from collaborative deep learning”, In Proc. of the ACM SIGSAC Conference on Computer and Communications Security, pp. 603–618, October 2017.

[19] Mäenpää, Dylan. "Towards Peer-to-Peer Federated Learning: Algorithms and Comparisons to Centralized Federated Learning." (2021).

[20] Jakub Koneˇcn, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. “Federated learning: Strategies for improving communication efficiency”.

[21] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." Artificial Intelligence and Statistics. PMLR, 2017.

[22] S. Kumar, S. Dutta, S. Chatturvedi and M. Bhatia, "Strategies for Enhancing Training and Privacy in Blockchain Enabled Federated Learning," 2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM), New Delhi, India, 2020, pp. 333-340

[23] J. Konen, H.B. McMahan, D. Ramage, and P. Richtrik, ”Federated optimization: Distributed machine learning for on-device intelligence,”

[24] V. Smith, C.K. Chiang, M. Sanjabi, and A.S. Talwalkar, ”Federated multi-task learning,” In Advances in Neural Information Processing Systems,2017 ,pp. 4424-4434.

[25] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Huba, V. Ivanov, C. Kiddon, J. Konecny, S. Mazzocchi, H.B. McMahan, and T. Van Overveldt, ”Towards Federated Learning at Scale: System Design.”

[26] G.J. Mendis, M. Sabounchi,J. Wei, and R. Roche, ”Blockchain as a Service: An Autonomous, Privacy Preserving, Decentralized Architecture for Deep Learning.”

[27] L. Feng, Z. Yang, S. Guo, X. Qiu, W. Li and P. Yu, "Two-Layered Blockchain Architecture for Federated Learning over Mobile Edge Network," in IEEE Network.

[28] K. Salah et al., “Blockchain for ai: Review and open research challenges,” IEEE Access, vol. 7, pp. 10127–10149, 2019.

[29] X. Zhang and X. Chen, “Data security sharing and storage based on a consortium blockchain in a vehicular ad-hoc network,” IEEE Access, vol. 7, pp. 58241–58254, 2019.

[30] J. Kang et al., “Blockchain for secure and efficient data sharing in vehicular edge computing and networks,” IEEE Internet of Things Journal, vol. 6, pp. 4660–4670, June 2019.

[31] Y. Lu, X. Huang, K. Zhang, S. Maharjan and Y. Zhang, "Blockchain Empowered Asynchronous Federated Learning for Secure Data Sharing in Internet of Vehicles," in IEEE Transactions on Vehicular Technology, vol. 69, no. 4, pp. 4298-4311, April 2020.

[32] R. Doku, D. B. Rawat, and C. Liu, “Towards federated learning approach to determine data relevance in big data,” in 2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI), pp. 184–192, July 2019.

[33] S. Samarakoon et al., “Federated learning for ultrareliable low-latency v2v communications,” in 2018 IEEE Global Communications Conference (GLOBECOM), (Abu Dhabi, United Arab Emirates), pp. 1–7, 2018.

[34] X. Bao et al., “Flchain: A blockchain for auditable federated learning with trust and incentive,” in 2019 5th International Conference on Big Data Computing and Communications (BIGCOM), (QingDao, China), pp. 151–159, Aug 2019.

[35] Kim, Hyesung & Park, Jihong & Bennis, Mehdi & Kim, Seong-Lyun. (2019). Blockchained On-Device Federated Learning. IEEE Communications Letters. 24. 10.1109/LCOMM.2019.2921755.

[36] U. Majeed and C. S. Hong, “Flchain: Federated learning via mec-enabled blockchain network,” in 2019 20th AsiaPacific Network Operations and Management Symposium (APNOMS), (Matsue, Japan), pp. 1–4, 2019.

[37] ark, S.; Suh, Y.; Lee, J. FedPSO: Federated Learning Using Particle Swarm Optimization to Reduce Communication Costs. Sensors 2021, 21, 600.

[38] Xu, J., Wang, H., & Chen, L. (2021). Bandwidth Allocation for Multiple Federated Learning Services in Wireless Edge Networks.

[39] W. Y. B. Lim et al., "Federated Learning in Mobile Edge Networks: A Comprehensive Survey," in IEEE Communications Surveys & Tutorials, vol. 22, no. 3, pp. 2031-2063, thirdquarter 2020.

[40] Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A. & Pedarsani, R.. (2020). FedPAQ: A Communication-Efficient Federated Learning Method with Periodic Averaging and Quantization. Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, in Proceedings of Machine Learning Research 108:2021-2031

[41] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng and Q. Yan, "A Blockchain-Based Decentralized Federated Learning Framework with Committee Consensus," in IEEE Network, vol. 35, no. 1, pp. 234-241, March/April 2021.

[42] Paritosh Ramanan and Kiyoshi Nakayama. BAFFLE : Blockchain Based Aggregator Free Federated Learning. 2019.

[43] Xiao, Yang, et al. "A survey of distributed consensus protocols for blockchain networks." IEEE Communications Surveys & Tutorials 22.2 (2020): 1432-1465.

[44] E. Bagdasaryan, A. Veit, Y. Hua, et al., How to backdoor federated learning, ArXiv Preprint ArXiv:1807.00459, August 2019. https://arxiv.org/abs/1807.00459 20, 140

[45] H. Kim, J. Park, M. Bennis, and S. Kim, “Blockchained on-device federated learning,” IEEE Communications Letters, vol. 24, no. 6, pp.1279–1283, 2020

[46] S. Otoum, I. Al Ridhawi, and H. Mouftah, “Blockchain-supported federated learning for trustworthy vehicular networks,” 12 2020, pp.1–6.

[47] Ma, Chuan, et al. "When federated learning meets blockchain: A new distributed learning paradigm." arXiv preprint arXiv:2009.09338 (2020).

[48] M. Jelasity, “Gossip,” in Self-organising Software. Springer, 2011, pp.139–162.

[49] Cachin, Christian, and Marko Vukolić. "Blockchain consensus protocols in the wild."

[50] . Qu et al., “Decentralized privacy using blockchain-enabled federated learning in fog computing,” IEEE Internet Things J., vol. 7, no. 6, pp. 5171–5183, Jun. 2020.

[51] Q. Wang, Y. Guo, X. Wang, T. Ji, L. Yu, and P. Li, “AI at the edge: Blockchain-empowered secure multiparty learning with heterogeneous models,” IEEE Internet Things J., vol. 7, no. 10, pp. 9600–9610, Oct. 2020.

[52] P. C. M. Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe, and M. Atiquzzaman, “A trustworthy privacy preserving framework for machine learning in industrial IoT systems,” IEEE Trans. Ind. Informat., vol. 16, no. 9, pp. 6092–6102, Sep. 2020.

[53] S. Lugan, P. Desbordes, E. Brion, L. X. R. Tormo, A. Legay, and B. Macq, “Secure architectures implementing trusted coalitions for blockchained distributed learning (TCLearn),” IEEE Access, vol. 7, pp. 181789–181799, 2019.

[54] H. Kim, J. Park, M. Bennis, and S.-L. Kim, “Blockchained on-device federated learning,” IEEE Commun. Lett., vol. 24, no. 6, pp. 1279–1283, Jun. 2020.

[55] S. R. Pokhrel and J. Choi, “Federated learning with blockchain for autonomous vehicles: Analysis and design challenges,” IEEE Trans. Commun., vol. 68, no. 8, pp. 4734–4746, Aug. 2020.

[56] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, “Low-latency federated learning and blockchain for edge association in digital twin empowered 6G networks,” IEEE Trans. Ind. Informat., vol. 17, no. 7, pp. 5098–5107, Aug. 2021.

[57] N. Q. Hieu, T. T. Anh, N. C. Luong, D. Niyato, D. I. Kim, and E. Elmroth, “Resource management for blockchain-enabled federated learning: A deep reinforcement learning approach,” May 2020. [Online]. Available: arXiv:2004.04104.

[58] H. T. Nguyen, N. C. Luong, J. Zhao, C. Yuen, and D. Niyato, “Resource allocation in mobility-aware federated learning networks: A deep reinforcement learning approach,” in Proc. IEEE 3rd World Forum Internet Things (WF-IoT), 2020, pp. 1–6.

[59] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, “Communication-efficient federated learning for digital twin edge networks in industrial IoT,” IEEE Trans. Ind. Informat., early access, Jul. 21, 2020.

[60] K. Toyoda and A. N. Zhang, “Mechanism design for an incentiveaware blockchain-enabled federated learning platform,” in Proc. IEEE Int. Conf. Big Data (Big Data), Dec. 2019, pp. 395–403.

[61] M. H. U. Rehman, K. Salah, E. Damiani, and D. Svetinovic, “Towards blockchain-based reputation-aware federated learning,” in Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS), Jul. 2020, pp. 183–188.

[62] N. B. Somy et al., “Ownership preserving AI marketplaces using blockchain,” in Proc. IEEE Int. Conf. Blockchain (Blockchain), Jul. 2019, pp. 156–165.