Blockchain-based Federated Learning with Secure Aggregation in Trusted Execution Environment for Internet-of-Things

Aditya Pribadi Kalapaaking, Ibrahim Khalil, Mohammad Saidur Rahman, Mohammed Atiquzzaman, Xun Yi, and Mahathir Almashor

Abstract—This paper proposes a blockchain-based Federated Learning (FL) framework with Intel Software Guard Extension (SGX)-based Trusted Execution Environment (TEE) to securely aggregate local models in Industrial Internet-of-Things (IIoTs). In FL, local models can be tampered with by attackers. Hence, a global model generated from the tampered local models can be erroneous. Therefore, the proposed framework leverages a blockchain network for secure model aggregation. Each blockchain node hosts an SGX-enabled processor that securely performs the FL-based aggregation tasks to generate a global model. Blockchain nodes can verify the authenticity of the aggregated model, run a blockchain consensus mechanism to ensure the integrity of the model, and add it to the distributed ledger for tamper-proof storage. Each cluster can obtain the aggregated model from the blockchain and verify its integrity before using it. We conducted several experiments with different CNN models and datasets to evaluate the performance of the proposed framework.

Index Terms—Federated Learning, Internet-of-Things, Blockchain, Secure Aggregation, Intel SGX, Trusted Execution Environment, Deep Learning

I. INTRODUCTION

The Internet-of-Things (IoT) explosion has made it an integral component of various intelligent applications. Intelligent applications include but are not limited to healthcare, manufacturing, critical system infrastructure, agriculture, and transportation. IoT devices enable the collection of a large volume of data and act autonomously in an intelligent system, thanks to machine learning algorithms. The large volume of IoT data plays an essential role in training a machine learning algorithm system. In general, IoT devices are resource-constrained and cannot execute machine learning algorithms independently. Edge computing technology is gaining acceptance at a tremendous rate to form intelligent networks in conjunction with IoT and machine learning. An edge device (referred to as an edge server throughout the article) and IoT devices within the network form a cluster. In an intelligent system, edge devices can host a machine learning algorithm that uses a locally-built dataset and produce a trained model. IoT devices generate data and receive control instructions depending on the type of IoT device. Later, the trained model can be used to make an intelligent decision in the system.

Although an edge and IoT-based system configuration with machine learning capability can manage different system tasks automatically, the level of accuracy impedes its success. For example, a trained model produced by an edge server with local data might not consider many features that could be absent in the local dataset. The accuracy can be improved if the edge device can collaborate with other edge servers that have produced their trained model based on their local datasets. This learning method is called *Distributed Collaborative Machine Learning*[1]. Traditional distributed collaborative machine learning (see Fig. 1) allows different clusters to send their locally-trained model and datasets to a centralized server, such as the cloud. Cloud aggregates all locally-trained models using datasets from different sources and produces an aggregated trained model shared with all clusters to improve decision-making accuracy.
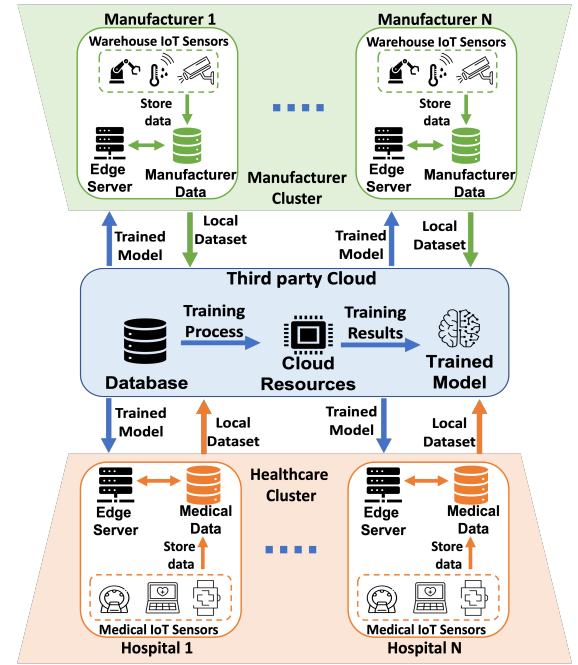

Fig. 1: Traditional collaborative learning application scenario

Distributed collaborative learning suffers from two significant issues: privacy and trust [2]. A new form of distributed collaborative learning, called *Federated Learning (FL)* [3], enables different clusters to build a trained model with their local data, called a *local model*, and to share only the local model with other participants for the purpose of aggregation. The aggregated model is known as a *global model*. Data privacy is ensured because the global model is generated without the data being shared with other participants. Nevertheless, the global model cannot be fully trusted as internal or external

attackers can launch several security attacks during the model aggregation and the dissemination of the global model. Hence, a trustworthy framework is required to ensure the privacy of sensitive data and the trustworthiness of the generated global model. Moreover, the receiver of the global model (e.g., edge server) should verify the integrity of the global model before using it.

A. Contributions

In this paper, we propose a complete framework for FL that simultaneously safeguards the privacy of IoT data and ensures security during the generation of aggregated trained models. In addition, the proposed framework guarantees trustworthy storage and sharing of the outcomes of any training. The proposed framework comprises a Convolutional Neural FL architecture that combines an Intel Software Guard Extension (SGX)-based Trusted Execution Environment (TEE) and blockchain platform. We assume that multiple IoT and edge devices clusters produce locally-trained models based on their local dataset and send the local model to the blockchain network for aggregation. In this framework, each blockchain node hosts an SGX-enabled processor that individually performs the FL-based aggregation tasks to generate an aggregated model. Once SGX-enabled processors of blockchain nodes perform the aggregation, each node can verify the authenticity of the aggregated model, run a blockchain consensus mechanism to ensure the integrity of the model, and add it to a blockchain for tamper-proof storage. An edge server from each cluster can collect the latest aggregated model from the blockchain and verify its integrity before using it. The key contributions of our work are summarized below.

- The proposed framework introduces a new FL architecture for IoT to ensure secure generation of the aggregated model using Intel SGX-powered TEE.
- We propose the hosting of an SGX processor by a blockchain node that is responsible for the FL model aggregation task.
- A blockchain-powered trustworthy aggregated model storage and sharing model is proposed for FL-based learning in IoT applications.

B. Organization

The rest of the paper is organized as follows. Section II describe the problem scenario in collaborative learning. Section III discuss some of the closely related work. The proposed framework is described in Section IV. Section V presents the experimental results and evaluates various performance aspects of the proposed framework. Section VI concludes the paper.

II. PROBLEM SCENARIO

To demonstrate and discuss the problem that exists with traditional collaborative machine learning, we use an IoT-enabled smart warehouse scenario (see Fig. 2). Assume that several smart warehouses are geographically dispersed. Each warehouse receives multiple pre-packed boxes of various garments (for both men and women), including shirts, trousers, shoes, jackets, and bags for storage. Each warehouse uses machine learning and an IoT-enabled camera to automatically sort the boxes according to the type of garment they contain. The camera scans the generic photo of the garment,

which is shown on the box. However, IoT-enabled cameras are resourced-constrained and cannot execute the machine learning algorithm. Hence, each warehouse is equipped with an edge server with access to the local dataset and hosts the machine learning algorithm to train a model for recognizing garment items based on the local dataset. Nevertheless, the accuracy of a training model derived from the local dataset may not be good. Therefore, the edge server of each warehouse participates in a cloud-based collaborative machine learning platform to share its local dataset and the trained model. The cloud-based collaborative machine learning platform produces an aggregated model based on the received local datasets and models. The aggregated model is sent to all edge servers to achieve higher accuracy in recognizing the garment items.

Although the aforementioned collaborative learning scenario improves overall accuracy, it suffers from the following security risks:

- *Risks of data privacy:* Sending local datasets to the cloud introduces the risk of a privacy breach. For example, a dishonest employee from the cloud service provider can act as an *internal attacker* and collect the warehouse's sensitive product information and share it with a business competitor for financial gain. Hence, there is the need for an aggregation model that would not require local datasets to generate an aggregated model.
- *Risks of generating biased aggregated trained model:* The aggregated model produced by a cloud service provider can be biased, as a cloud-based platform cannot be trusted. For instance, an internal attacker can generate a biased aggregated model not using the given local models or inject a faulty trained model to interrupt the generation of aggregated models. Therefore, a secure environment is required to prevent biased model generation.
- *Risks of receiving alteration or faulty aggregated trained model:* In the traditional cloud-based collaborative learning environment, an internal attacker of the cloud platform can interfere with disseminating the aggregated model. For example, an attacker can alter some part of the aggregated model before the cloud sends it to the edge servers. The traditional method does not allow a receiver of the aggregated model (i.e., edge server) to verify its integrity before using it. Hence, a trustworthy platform is required for sharing the aggregated model with edge servers.

III. RELATED WORK

This section discusses several studies that are closely related to our work.

Privacy-preserving Federated Learning. Several works on privacy-preserving federated learning have been presented recently. Yin *et al.* [4] and Liu *et al.* [5] proposed a federated learning framework where the training is performed on each node and only the model is sent to the central server to perform the model aggregation. Wei *et al.* [6] and Zhao *et al.* [7] proposed a framework where data privacy is improved by means of differential privacy. However, the use of DP will slow down the training process and reduce accuracy. In [8] the author proposes anonymous federated learning by adding
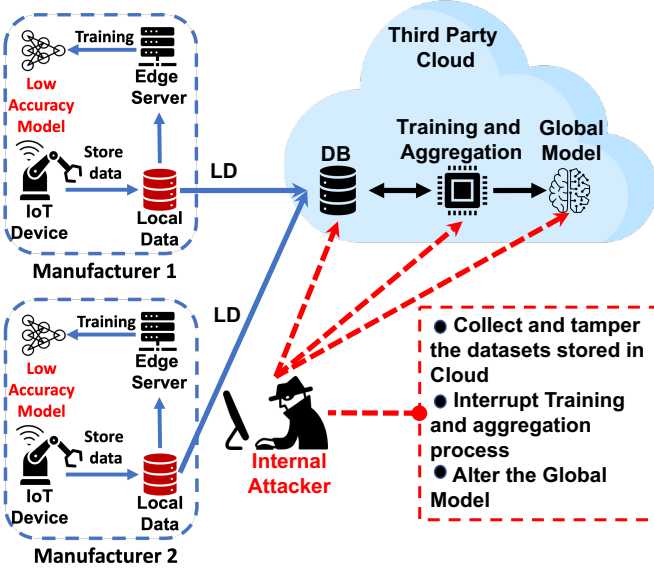
Fig. 2: Possible threat on collaborative learning architecture

computational nodes in a distributed fashion. Blockchain is structured as a linked list of blocks holding a set of transactions. Ali *et al.* [16] proposed a method to ensure the privacy and security of healthcare systems using blockchain. Their approach focuses mainly on securing patient data from active collision attacks by leveraging novel smart contracts and encryption algorithms. Nowadays, many studies are incorporating blockchain into their federated learning methodologies because federated learning is based on a centralized server, which is vulnerable to attack. Zhao *et al.* [17] designed a system where each of the clients will sign the model after the training process and send it to the blockchain. However, if this model has many clients, the computation cost will be very high. In recent works, [18], [19], and [20] proposed a framework where the model is stored in the blockchain node, and federated learning is performed. However, in their architecture, the model is not totally encrypted. Also, the aggregation is performed by an untrusted party. Kim *et al.* [21] proposed a method where they deploy the blockchain on the edge devices. The disadvantage of this method is that the edge devices will require a lot of computation power. The author in [22] proposes a blockchain architecture to collect the locally-trained model weights collaboratively from different sources for healthcare scenarios. However, the local model that is stored in the blockchain is not protected by any privacy measure. In this case, other parties can see the model, thereby raising privacy issues.

Samuel *et al.* [23] proposed blockchain-based FL for healthcare system. Their proposed framework protects the local model training with differential privacy (DP). The central server aggregates the global model and stores it in the blockchain. However, the global model accuracy is lower than the locally trained model. The use of DP in this framework can preserve privacy while sacrificing accuracy. Alsamhi *et al.* [24] and Otoum *et al.* [25] proposed an edge intelligence over smart environments with the support of FL and blockchain. Their proposed architectures leverage drones as an edge intelligence to perform the aggregation in FL. The aggregation process on a drone is vulnerable to tampering attacks and poisoning attacks. Since drones are deployed on the field and open networks, hardware security such as TEE can secure the aggregation process.

In Table I, we summarize some of the works to identify their research gaps and discuss how our proposed method differs from them. As shown in the table, existing works are mostly unsecured, inefficient, and have low accuracy. Hence, we deploy the blockchain on the server-side to reduce training model storage costs and leverage TEE to ensure secure and trustworthy model aggregation before sending it to the blockchain.

a proxy layer and DP to the data. However, the proxy layer will add communication overhead, and the result shows that the DP decreases the ML accuracy. Li *et al.* [9] leverage SMPC-based federated learning to secure aggregation. Hence, their framework relies on a centralized server to arrange the secret sharing. This could be a problem since all the models can be seen in plaintext after the cloud collects the secret share. Federated learning is delicate to an attacker that can launch backdoor attacks. Bagdasaryan *et al.* [10] found that a backdoor can compromise the federated learning and poison the machine learning model. Our framework will create a secure end-to-end federated learning process to overcome this problem by securing the machine learning model and the aggregation process.

TEE-based Machine Learning. Recently, TEE has gained popularity in the field of privacy-preserving machine learning. Ohrimenko *et al.* [11] investigated centralized machine learning processes in an SGX-enabled data center to improve data privacy and avoid data leaks. In his framework, the server requests the dataset from all the participants and computes it in a centralized server. Tramer *et al.* [12] and Juvekar *et al.* [13] proposed a secure inference process inside of the TEE. Hynes *et al.* [14] and Hunt *et al.* [15] demonstrated centralized privacy-preserving machine learning by running all the CNN processes inside the enclave.

The available frameworks use a single deep learning model, and none of them performs within the federated learning setup. The current work also shows that the time cost is significantly increased when the training process is performed in the TEE. Hence, we run the aggregation process inside the enclave to maximize the performance and reduce time consumption.

Blockchain-based Federated Learning. Blockchain was first launched as a cryptocurrency technology. However, it has now been expanded for data storage across multiple

IV. PROPOSED FRAMEWORK

In this section, we present the proposed blockchain-based federated learning with Trusted Execution Environment (TEE)-powered secure aggregation framework. First, we present an overview of the system architecture. Next, we discuss in detail the various components of our proposed framework.

Methodology	Description	Remarks
Yin *et al.* [4]	A privacy-preserving machine learning approach based on DP and sparse vector technique.	Low model accuracy and security is poor.
Wei *et al.* [6]	A FL approach with DP to secure the data on edge devices.	Centralized approach with lower accuracy and efficiency.
Ohrimenko *et al.* [11]	TEE-based machine learning approach.	Centralized approach with lack of data privacy. TEE is leveraged during local training. Hence, may not be suitable for Edge devices.
Hunt *et al.* [15]	TEE-based privacy-preserving Machine Learning approach	Secure model generation through TEE; however, the global model is generated and stored in the unsecure centralized server.
Qu *et al.* [26]	A blockchain-based FL approach. Blockchain is used to store model securely. Edge for cognitive computing in industrial IoT. Their framework keeps the local data on their edge devices and uses blockchain to secure the model.	The model aggregation is performed in an untrusted environment; hence, susceptible to tampering attacks.
Kim *et al.* [21]	A blockchain of edge devices and FL based fully decentralized approach.	Inefficient and the blockchain-based design is too heavy to implement in edge devices.

TABLE I: Summary of Related Works

TABLE II: Notations

M_L	Local Model
M_G	Global Model
M_{Li}^{r+1}	Updated Local Model
M_{Gi}^{r+1}	Updated Global Model
D_i	Local Image Dataset
C_i	IoT Cluster
E_i	Edge Server
B_i	Blockchain Node
S_i	SGX-enabled CPU
$E(M_{Li}, K_i)$	Encrypted Training model
R_i	Remote Attestation Report
Q	Quotation for Global Model

A. System Architecture

We consider a Federated Learning (FL)-based collaborative learning model in this system that leverages Trusted Execution Environment (TEE) for secure aggregation and blockchain for tamper-proof data sharing and storage.

We assume that there are p warehouses equipped with several IoT cameras to scan product photos and recognize the type of products. Because IoT cameras are resource-constrained when running machine learning algorithms, each warehouse uses an edge server to host and execute a machine learning algorithm. As a result, IoT cameras and the edge server form a cluster $C_i(1 \leq i \leq p)$. Initially, the edge server trains a model based on the local dataset and generates a trained model called a *Local Model*, denoted as M_L. However, if the size of the local dataset is small, the accuracy of M_L might not be high. Hence, the edge server of a cluster C_i joins in FL involving multiple clusters of similar warehouses by sending its M_L to generate an aggregated model known as a *Global Model*, denoted as M_G. In our proposed scenario, we adopt Federated Averaging (FedAVG) [27] algorithm for generating the global model, which will be discussed in Section IV-C.

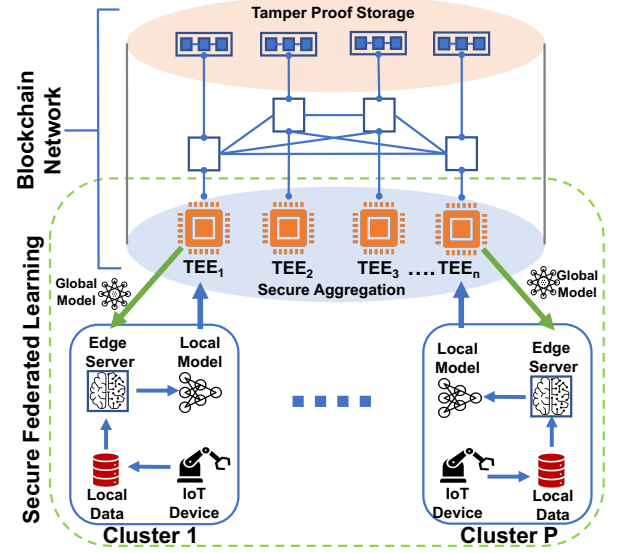A typical FL approach involves three steps: *initialization*,

Fig. 3: Overview of the proposed framework

aggregation, and *update*. Unlike the traditional FL approach where M_L are aggregated in a centralized server (e.g., a cloud server), our proposed framework uses a blockchain platform for the aggregation of M_L. Multiple nodes form a blockchain network, and each node receives all M_L and individually aggregates M_L to produce their own copy of a M_G. We assume that each blockchain node has a *TEE host*. To ensure the security during the aggregation process, each blockchain node performs the aggregation in its TEE host and produces a M_G. Blockchain nodes execute a consensus mechanism to ensure that all nodes have identical M_G. Once the consensus has been reached, each blockchain node stores the M_G in its respective blockchain. Finally, the blockchain network sends M_G to all edge servers. Edge servers validate M_G once received and update their initial model with M_G. Edge servers use the M_G for product recognition in the warehouse.

Fig. 3 gives an overview of the proposed framework, which consists of three main phases: *Local Model Generation*, *Secure TEE-Enabled Aggregation*, and *Blockchain-Based Global Model Storage*. The following subsections describe each phase in detail.

B. Local Model Generation

The Local Model Generation (LMG) phase is performed in each cluster to generate a locally-trained model similar to the initialization phase of the original FL. An overview of the LMG phase is given in Fig. 4. In the proposed system, we assume that the edge servers of different clusters train models using Convolutional Neural Network (CNN)-based image classification in which model parameters are retrieved from the global model stored in the tamper-proof storage. Example of the CNN models are AlexNet[28], LeNet[29] and VGG16[30].

In general, CNN image classification takes an input image, processes it and classifies it under certain categories of t objects. An edge server E_i of cluster C_i has a local image dataset D_i. The edge server sees an input image as an array of pixels, and it depends on the image resolution. Based on
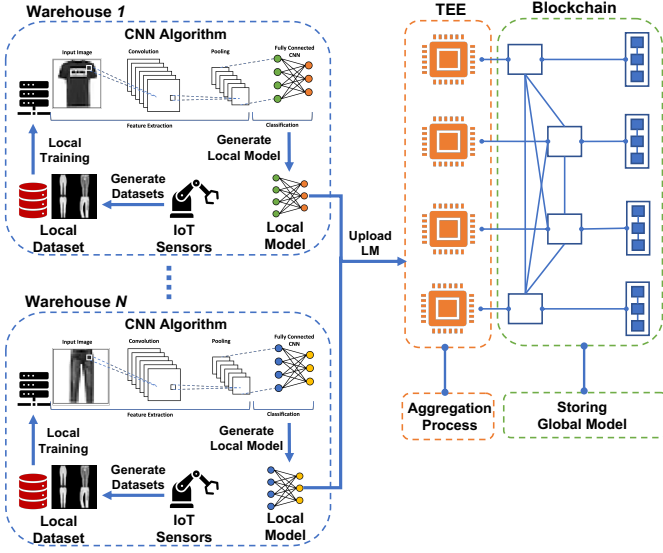
Fig. 4: Local model generation

the image resolution, it will see $h \times w \times d$ (h = Height, w = Width, d = Dimension). For example, an image of 6 x 6 x 3 array of a matrix of RGB (3 refers to RGB values) and an image of 4 x 4 x 1 array of a matrix of a grayscale image. Technically, the deep learning CNN model works via different layers to train and test a local model. The layers are *convolution layers with filters (kernels)*, *pooling*, and *fully connected layers (FC)*. In the end, CNN applies the *SoftMax function* to classify an object according to probabilistic values between 0 and 1. In each edge server, E_i locally trained the machine learning model M_{Li}. An edge server E_i updates the ML model using its dataset in every FL round r as follows:

$$M_{Li}^{r+1} = M_G^r - \eta \nabla F(M_G^r, D^i) \tag{1}$$

where M_{Li}^{r+1} denotes the updated local model of client i, M_G^r is the current global model, η is the local learning rate, ∇ is used to refer to the derivative with respect to every parameter, and F is the loss function. Later, E_i send M_{Li}^{r+1} to the blockchain network and aggregated iteratively into a joint global model M_G.

To ensure the security of the local model, E_i leverages symmetric key encryption algorithm, such as Advanced Encryption Standard (AES), to encrypt M_{Li} before sending it to the blockchain nodes. We assume that the AES secret key between E_i and the blockchain network is established using a secure key establishment mechanism, such as Diffie–Hellman key exchange mechanism. We do not discuss this process in detail as we leverage the state-of-the-art mechanism for encrypting the local model.

C. TEE enabled Secure Model Aggregation

Once different local models are received by a blockchain node, a TEE is used to securely aggregate all models. For TEE, we use Intel Software Guard Extension (SGX) [31] in this framework. SGX is a set of CPU extensions, which can provide isolated execution environments, named *enclaves*, to protect the confidentiality and integrity of the data against

all other software, even a compromised OS, on the platform. When a platform is equipped with an SGX-enabled CPU (such as a blockchain node in our framework), as an enclave, the memory, BIOS, I/O, and even power are treated as potentially untrustworthy. Firstly, the encrypted data is transmitted into an enclave for decryption. Then the decrypted data will be the input of function f. Finally, the output of f will be encrypted and then sent to the outside of the enclave.

Using the same principle, FL's local model aggregation task is performed in the SGX-enabled CPU. We assume that there are b blockchain nodes in the blockchain network, and each blockchain node $B_i (1 \le i \le b)$ in the blockchain network is equipped with an SGX-enabled CPU S_i. A blockchain node B_i cannot access the code and data within its SGX-enabled CPU S_i.

Assume that a blockchain node B_i receives the set M_L of local models from all clusters which can be denoted as $M_L = \{M_{L1}, M_{L2}, \ldots, M_{Lp}\}$. B_i sends $M_{Li}(1 \le i \le C_i)$ to S_i. The secure aggregation tasks of all local models in M_L is done using multiple operations which are discussed below.

1) Generation of Encrypted Local Models: The SGX enclave receives only encrypted data to ensure security. Hence, B_i needs to encrypt M_{Li} before sending it to S_i. Let, $E(., K)$ be a Symmetric Encryption (SE) algorithm $E(., K_i)$ with a secret key K_i that is shared between B_i and its S_i. The shared secret key K_i is established by leveraging a secure key exchange protocol such as Diffie-Hellman Key Exchange Protocol.

B_i generates an encrypted local model $E(M_{Li}, K_i)$. B_i sends $E(M_{Li}, K_i)$ to S_i.
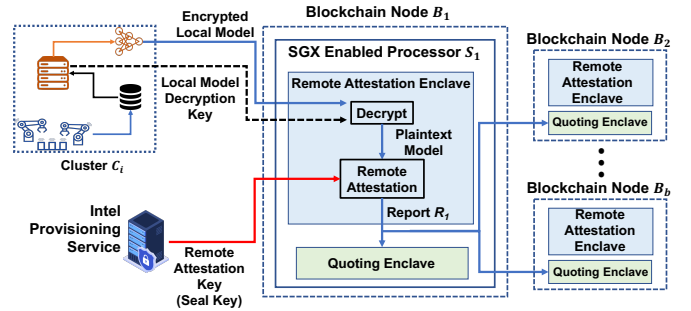

Fig. 5: Remote attestation of local model

2) Remote Attestation: The remote attestation allows the verification of the integrity of the aggregated model (i.e., global model) generated by the SGX enclave. In this framework, the SGX enclave acts as the *attestator*, and the software module of a blockchain node B_i that is responsible for interfacing between SGX enclave and the blockchain network plays the role of a *verifier* of the attestation. *First*, the SGX enclave S_i receives the set M_L^E of p encrypted local models which can be denoted as $M_L^E = \{E(M_{L1}, K_1), \ldots, E(M_{Lp}, K_p)\}$. S_i decrypts each $E(M_{Li}, K_i)(1 \le i \le p)$ with the shared secret key k_i to retrieve the plaintext set of local learning models M_{Li}. *Second*, S_i performs the aggregation using the Federated Averaging (FedAVG) [27] algorithm to generate the global model M_G as follows:

$$M_{Gi}^{r+1} = \sum_{i=1}^{n} \frac{|D_i|}{N} M_{Li}^{r+1}, N = \sum_{i=1}^{n} |D_i| \qquad (2)$$

where M_{Gi}^{r+1} denotes the updated global model, n is a number of clients on the federated learning round r, $|D_i|$ is the number of data items (images) owned by E_i to train local model M_{Li}^{r+1}, and N the total number of data used to train all of the local models. M_G is final updated global model M_{Gi}^{r+1}.

Third, S_i generates a remote attestation, called *report* $R_i = Sign(M_{Gi}, A_{Ki})$. Here, $Sign(., A_{Ki})$ is a signature function and A_{Ki} is the attestation key of S_i. The generated report enables a verifier (i.e., blockchain node) to verify the M_{Gi}. The pseudocode of the overall aggregation and remote attestation is illustrated in Algorithm 1. The algorithm takes encrypted trained models as input and outputs aggregated global models, and its remote attestation report All tasks of Algorithm 1 are executed under a running enclave of the SGX. SGX uses a quoting enclave to verify reports produced by the application enclave and signs as a quote. The quoting enclave is used to determine the trustworthiness of the platform. Later, the quote is sent to another party for verification. In our scenario, each B_i will have one S_i and works as an aggregator and verifier of attestation reports. Fig. 5 shows the details of the quoting enclave process.

Algorithm 1: Aggregation Process and Remote Attestation in TEE

Input:
    Encrypted Trained Models $M_L^E = \{E(M_{L1}, K_1), \ldots, E(M_{Lp}, K_p)\}$

Output:
    Aggregated Global Model (M_G) and
    Remote Attestation Report R_i

1 **while** *SGXServerRunning* **do**
2   **while** *EnclaveRunning* **do**
3   **Initialize:**
4   Memory Buffer, $Mem = \emptyset$
5   **for each** $E(M_{Li}, K_i) \in M_L^E$ **do**
6   Decrypt $E(M_{Li}, K_i)$ with the key K_i to obtain the corresponding decrypted local model $\overline{M_{Li}}$.
7   Add $\overline{M_{Li}}$ to memory buffer Mem.
8   **endForEach**
9   Aggregate all decrypted local models in Mem according to **FedAvg** algorithm as shown in (3) and generate global model M_G.
10   Generate a remote attestation R_i for M_G by signing it with the attestation key A_{Ki} of S_i.
11   **return** $\{M_G, R_i\}$
12   **endWhile**
13 **endWhile**

D. Blockchain-based Tamperproof Global Model Storage and Distribution

In this phase, the blockchain network receives all remote attestations produced by SGX enclaves and runs a consensus mechanism. The consensus mechanism verifies the remote attestations of a global model produced by the SGX enclaves. If all remote attestations are verified, and the majority hashes of corresponding models are the same, the blockchain nodes in
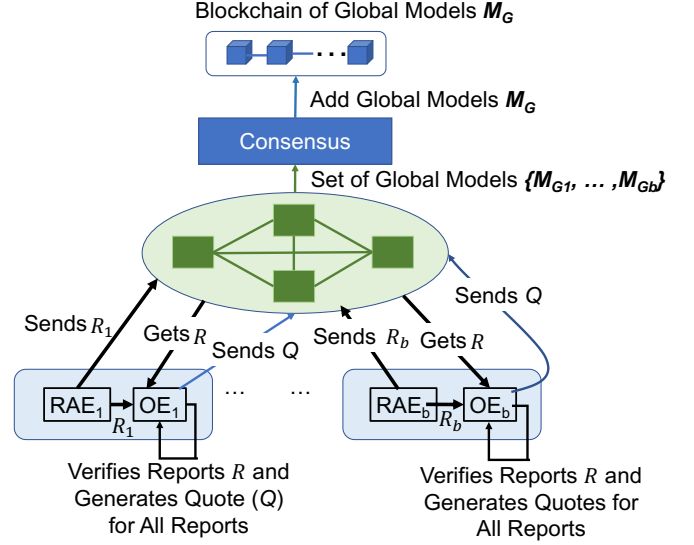

Fig. 6: Blockchain based global model storage

the blockchain network add the global model M_G as a block in the blockchain. Also, the global model is sent to all edge servers as the update operation FL. An overview of this step is given in Fig. 6.

1) Verifying Attestation Reports by a Blockchain Node: Assume that each blockchain node is equipped with a *quoting enclave* and has an attestation key A_{Kj} to sign a remote attestation report R_i produced by S_i. R_i is signed with A_{kj} to generate a *quote* Q_i. A quote contains the identity of the attesting enclave S_i, execution mode details (e.g. Security Version Number level S_i), and additional metadata. The function that is used to generate Q_i can be shown as: $Q_i = Sign(R_i, A_{Kj})$. Q_i is encrypted using the public key PK_{IAS} of Intel Attestation Service (IAS) and generates $E(Q_i, PK_{IAS})$. PK_{IAS} is embedded in the quoting enclave of all SGX-enabled processors. Each S_i shares its $E(Q_i, PK_{IAS})$ to other SGX-enabled processors of the blockchain network. Once all encrypted quotes are received from SGX-enabled processors of all n blockchain nodes, B_i creates a collection of Encrypted Quotes received from all which is denoted by $Q^E = \{E(Q_1, PK_{IAS}), E(Q_2, PK_{IAS}), \ldots, E(Q_n, PK_{IAS})\}$.

A blockchain node B_i verifies each encrypted quote with the help of IAS and determines if the quote is correct and the corresponding remote attestation enclave has created it. The verification is done using a function $verify(E(Q_i, PK_{IAS}), PR_{IAS})$, where PR_{IAS} is the private key of IAS. Once the quotation is verified, Q is broadcast to the blockchain network to obtain a consensus for the global model. Algorithm 2 provides an overview of this step.

2) Consensus by Blockchain Network: The consensus mechanism has several steps. *First*, a blockchain node B_i checks the validity of each quote and the authenticity of the quote-generating enclave. *Second*, the global model is extracted from each quote, and their hashes are verified. If the hashes of all global models are the same, the consensus is achieved. If all hashes are not the same, the blockchain node B_i determines the global model M_{Gk} that has maximum
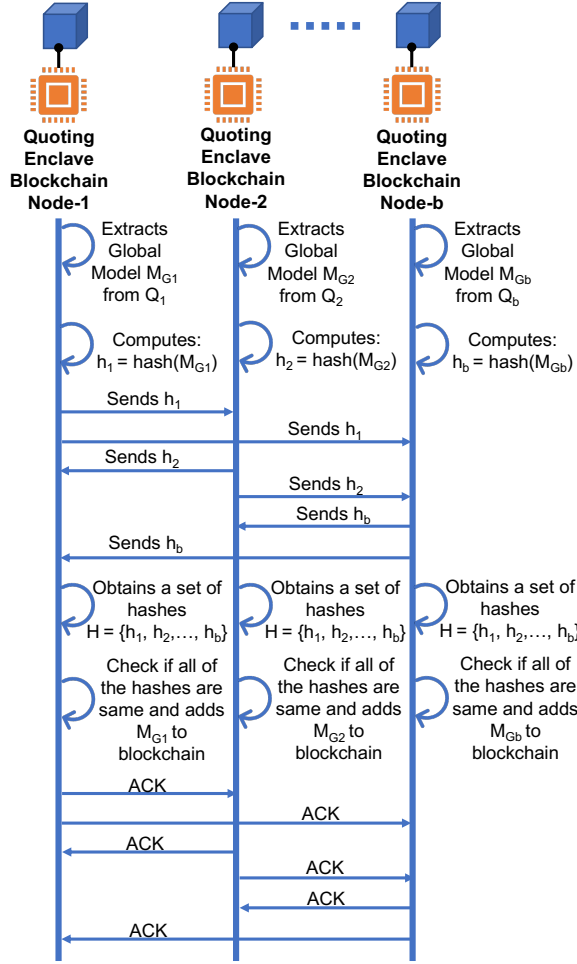
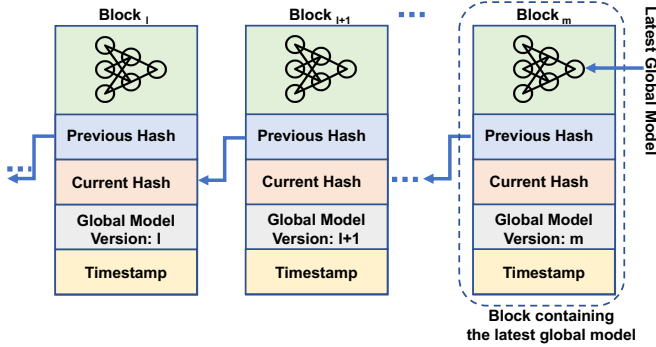**Fig. 7:** Overview of the consensus mechanism

Fig. 8: Data structure of the global model blockchain

matched hash values, where $k \leq p$. Third, B_i proposes M_{Gk} to the blockchain network to add in the blockchain. Finally, if M_{Gk} is the same for the majority of the node's global model, the consensus is achieved and added to the blockchain. Algorithm 3 shows the pseudocode and Fig. 7 provide an overview of of consensus mechanism. The blockchain data structure of global models is illustrated in Fig. 8.

Algorithm 2: Quote Generation for Global Model

Input:
    Remote Attestation Reports $R = \{(R_1, Pk_1), \ldots,$
    $(R_p, Pk_p)\}$

Output:
    Quotation for Global Model (Q)

1 **while** *SGX Server is Running* **do**
2   **while** *While Quoting Enclave is Running* **do**
3   Collection of Quotes, $Q = NULL$
4   **for each** $R_i \in R$ **do**
5   Verify the validity of the report R_i done by Enclave S_i
6   Generate a Quote, $Q_j = Sign(R_i, A_{kj})$
7        $Q.add(Q_j)$
8   **endForEach**
9   Broadcasts Q to Blockchain Network
10   **endWhile**
11 **endWhile**

Algorithm 3: Consensus Mechanism on Global Model

Input:
    Quotation for Global Model (Q)

Output:
    Global Model (M_G)

1 **while** *Blockchain Node B_i is Running* **do**
2   Set of Global Models, $GM = NULL$
3   Set of Hashes of Global Models, $H = NULL$
4   **for each** $Q_j \in Q$ **do**
5   Verify the validity of the Quote Q_i done by Enclave S_i
6   Extract global model M_{Gj} from Quote Q_j)
7   Add M_{Gj} to GM
8   Send M_{Gj} to other nodes in the blockchain network
9   **endForEach**
10   **for each** $M_{Gj} \in GM$ **do**
11   Get the hash h_j of M_{Gj} and it to H
12   **endForEach**
13   Add M_{Gj} to Blockchain if all hashes in H are same.
14 **endWhile**

V. RESULTS AND DISCUSSION

In this section, we report on several experiments conducted to evaluate the performance of our proposed framework. Experimental setup, and dataset and model are discussed in Section V-A and V-B, respectively. Section V-C shows experimental results and evaluates the performance.

A. Experimental Setup

In our experiments, both the server and participant applications were run on an Azure Cloud. We used the DCsv2 series VM with 4 vCPU and 16 GB Memory. This DC series from Azure provides confidentiality and integrity of the data and code while they are being processed in the public cloud. DCsv2-series using Intel® Software Guard Extensions was used, which enables the end-user to use secure enclaves for protection. These machines are backed by 3.7 GHz Intel® Xeon E-2288G (Coffee Lake) with SGX [31] technology. We built our federated learning application based on PyTorch [32] and PySyft [33]. To run the PyTorch application in the SGX environment, we build our application on GrapheneOS [34].

B. Datasets and Model

For the experiments, we selected three datasets popularly used for the machine learning process: Fashion MNIST [35], CIFAR-10 [36], and MNIST [37]. These datasets are commonly used for benchmarking in the machine learning framework. Therefore, we have used them to evaluate the performance of our proposed approach. The dataset is used to train and test the local model on the client-side in the proposed FL-based approach. For all our experiments, we split the training and testing sets. Based on the number of participants, we evenly distribute the training and test sets among all participants. Fashion MNIST [35] is a collection of datasets containing fashion images. The training set comprised 60,000, and 10,000 images were used as a test set. Each image had a 28×28-pixel grayscale, and nine different classes were represented (trousers, dress, bag, etc.). MNIST [37] is a dataset consisting of handwritten digits (60,000 images in the training set and 10,000 in the test set). Each image is a 28×28-pixel image of a handwritten digit. CIFAR-10 [36] consists of 50,000 images in the training set and 10,000 in the test set. It comprises 10 different classes (such as cars, dogs, planes), and there are 6,000 images in each class, where each image contains 32×32-colored pixels. Table III shows the overview of the dataset used in the experiments.

We consider three models for our experiment. First, the LeNet model was used, which was proposed by LeCun et al. [29]. The model contains two convolutional layers and two fully-connected layers. This model is suitable for running experiments using the Fashion MNIST and MNIST datasets. Second, the AlexNet [28] model is used with five convolutional layers and three fully-connected layers. This model can use batch normalization layers for stability and efficient training. AlexNet is suitable for testing on the CIFAR-10 datasets. Finally, the VGG16 [30] model is used that has 16 layers and about 138 million parameters. This machine learning model is also suitable for CIFAR 10 datasets.

Datasets	Training set	Test set	Size	Color
MNIST [37]	60.000	10.000	28x28	Grayscale
F-MNIST [35]	60.000	10.000	28x28	Grayscale
CIFAR-10 [36]	50.000	10.000	32x32	RGB

TABLE III: Datasets specifications

C. Experimental Results and Performance Evaluation

First, we evaluate the performance of our framework for the global model aggregation process(see Fig. 9). This experiment shows the time cost difference when performing the secure aggregation process with enclaves and without enclaves with various numbers of edge devices ranging from 2 to 40. Results show the aggregation times required by LeNet, AlexNet, and VGG16 for a batch size of 128. In Fig. 9a and Fig. 9d, we show the LeNet model with Fashion MNIST and MNIST dataset, respectively. The experimental results show that the LeNet model exhibits a similar trend when used on Fashion MNIST and MNIST datasets. The time is consistently stable when it has 20 edge devices, and the time cost rises a little bit when it reaches 30 edge devices during the aggregation in the
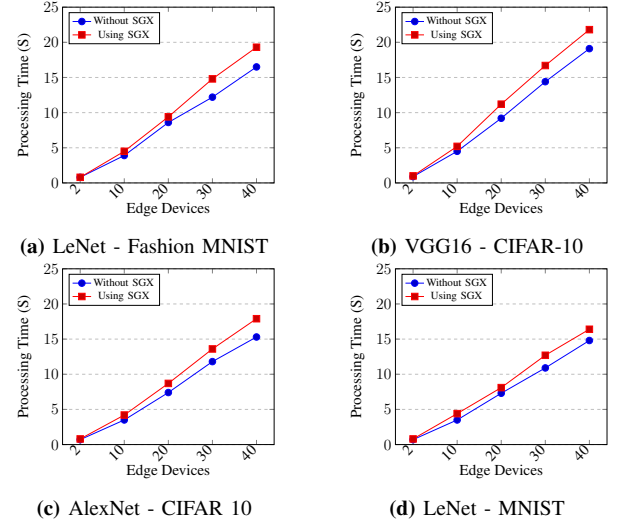

Fig. 9: Processing time of secure aggregation process with and without SGX using various machine learning models and datasets considering batch size = 128.

TEE. The average additional time cost is 1.2 seconds. Figures 9b and 9c show the results of AlexNet and VGG-16 models with CIFAR-10 dataset. When we perform the VGG-16 model with 40 edge devices, the aggregation process without SGX requires 19.1 seconds. The aggregation process is higher with SGX, which is 21.8 seconds. The required time to aggregate local training models of VGG-16 is the highest due to the involvement of 16 layers. Nevertheless, it is only 2.7 seconds slower than the time cost of aggregation without SGX. Fig. 9 shows that the aggregation time cost is 1.3 seconds higher on an average in SGX due to the paging mechanism and memory limitation of SGX.
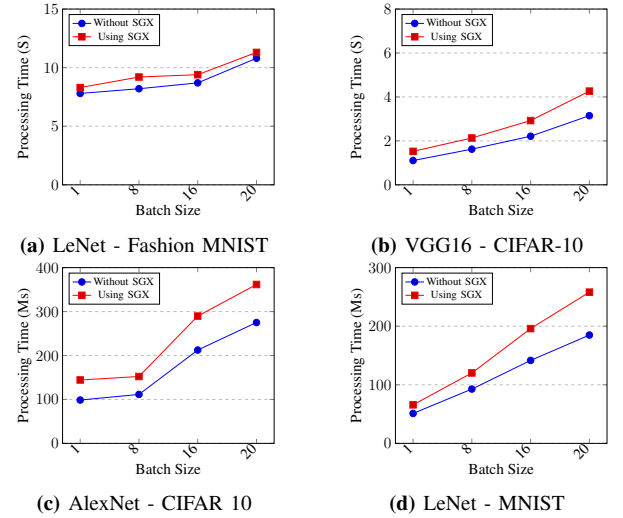

Fig. 10: Processing time of training process with and without SGX for different number of batch size using various machine learning models and datasets.

In Fig. 10 we test the performance of our framework using different machine learning models and datasets. The experiment is conducted within and outside the enclave with

different batch sizes (1, 8, 16, and 20), and the time costs of the training processes are shown in Fig. 10a. The time cost of the LeNet machine learning model with Fashion MNIST datasets running outside the enclave starts from 7.2 seconds for one batch size. The time cost increases linearly to 8.7 seconds for 16 batch size. Fig. 10b shows the time costs of VGG-16 with CIFAR 10 datasets. The time cost is 1.1 seconds for batch size is 1 and 2.2 seconds for batch size 16. The time costs increase slightly, keeping the same linear characteristics when the experiments are performed inside the enclave with the same settings. The LeNet model requires 8.1 seconds to 9.4 seconds, while VGG-16 requires 1.5 seconds to 2.9 seconds. The experiment results show that the time cost increases for both inside and outside enclave training when all the machine learning models use 20 batch sizes. The time costs also increase if the number of images in a batch increases.
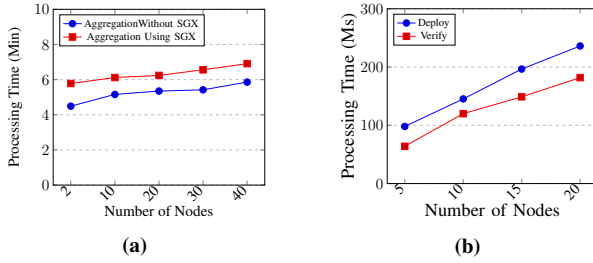

Fig. 11: Processing time: (a) for the federated learning process using LeNet model[29] with Fashion MNIST datasets[35] with different number of nodes, and (b) for adding the global model to the blockchain with different number of blockchain nodes.

Fig. 11a shows the required time for the FL process with different numbers of edge servers. In this experiment, we performed multiple federated learning processes that use normal aggregation and SGX-based secure aggregation. We used pre-processed the Fashion MNIST dataset and the LeNet machine learning model with 128 batch sizes for the experiment. We consider different number of edge servers ranging from 2 to 40. Results indicate that the time cost increases gradually for the federated learning process with and without SGX-based aggregation. When comparing the results of normal CPU and SGX based approach, the time differs about 70 milliseconds with 10 edge servers and 91 milliseconds with 40 edge servers.

Fig. 11b shows the time required to execute both the verification and deployment of the global model in our blockchain network. In this experiment, the TEE aggregates all the models from the edge server to form a global model. The global model is then verified and deployed in the blockchain network. Our simulation tested the performance using several blockchain nodes ranging from 5 to 20. The deployment phase requires roughly 100 milliseconds (with 5 blockchain nodes) to 230 milliseconds (with 20 blockchain nodes). The verification phase is faster than the deployment phase and requires 60 milliseconds and 180 milliseconds with 5 and 20 blockchain nodes, respectively. The processing times of both phases increase linearly with the increment of blockchain nodes.

Table IV shows the results of testing our framework to see the effect when we apply the machine learning model in a federated way inside the enclave and standard CPU. In

Methodologies	CNN Model	Dataset	Baseline	SGX	Accuracy Reduction
Proposed Method	LeNet[29]	F-MNIST[35]	93.2%	90.8	2.4%
Proposed Method	AlexNet[28]	CIFAR-10[36]	73.3%	70.4%	2.9%
Proposed Method	VGG-16[30]	CIFAR-10[36]	87.4%	84.8%	2.6%
Proposed Method	LeNet[29]	MNIST[37]	95.7%	93.6%	2.1%
Myelin[14]	RESNET-32[38]	CIFAR-10[36]	89.5%	84.4%	5.1%
Chiron[15]	VGG-9[30]	CIFAR-10[36]	88.5%	81.1%	7.4%

TABLE IV: Comparison of machine learning model accuracy in federated learning process when using normal CPU and SGX

this experiment, all the datasets have 28x28 pixels and 128 batch size. We ran the experiment with 50 training iterations. The experimental results show that the differences in the accuracy of the proposed methodology and two benchmark methods proposed in [14] and [15]. Initially, we record the accuracies of our proposed method with and without SGX. The accuracies of the aforementioned methods are obtained by applying various CNN models on different datasets. According to the results, the accuracies are reduced by 2.2% to 2.9% when SGX is used. Later, we measure the accuracies of Myelin[14] and Chiron[15] with and without SGX. Results show that accuracies of the Myelin and Chiron are lower than our proposed method. Moreover, the accuracies of Myelin and Chiron are reduced around 5.1% and 7.4% with SGX, respectively. Hence, our method has better accuracy compared to Myelin and Chiron.

D. Discussion

In this section, we summarize the performance of our proposed method. As discussed in Section V-C, we conducted a series of experiments to evaluate the efficacy of our proposed method. Based on the empirical results, the following conclusions can be drawn.

- **Privacy of Local Dataset**: Federated learning allows computational parties to collaboratively learn a shared model while preserving all training data locally, separating the machine learning process from the storage of data in the central server. The method is unlike traditional centralized machine learning where local datasets are stored in one central server. Therefore, federated learning can ensure the privacy of the client's sensitive data.
- **Privacy of Local Training Model**: In our framework, the local training model is encrypted using a shared key before it is sent to the blockchain node. The shared key is established using a secure key-exchange protocol. Later, the local training model will be decrypted inside the enclave for secure aggregation. As the local model is encrypted, model inversion attacks [39], and parameter stealing [40] cannot be performed on a local model by an adversary.
- **TEE-based Secure Aggregation**: In federated learning, aggregation is typically performed on a normal server. Several researchers [6], [7] have proposed a differential privacy (DP) method to secure the model during the aggregation process. However, DP will significantly reduce the accuracy of the global model. Table. IV shows that the use of secure TEE-based aggregation can overcome this problem while maintaining the privacy of the model.

As the aggregation is performed in the TEE, adversaries cannot tamper with or steal the model parameters during the aggregation process. As blockchain technology is being used with emerging technologies, such as drones[41], [42], [43], the proposed blockchain and TEE-based model aggregation in FL would enhance the trust in applications where drones are used as edge intelligence in FL [24], [25].

- **Resilience of the Global Model**: Blockchain is a decentralized technology that can maintain data integrity by means of an extensive network that can withstand security breaches from untrusted parties. In the proposed framework, we use blockchain to store the global model after the aggregation process in the TEE. This decentralization makes it almost impossible for an adversary to compromise the network. Moreover, model updates are protected by digital signatures and hashes. Hence, the adversary cannot tamper with or contaminate the global model since this will change the hash value.

- **Model Performance**: Although our proposed method can ensure the privacy of the model and the security of the aggregation process, performance is still a crucial metric for measuring the quality of the framework. The experimental results show that the performance of the proposed framework is better than that of the baseline model. Our proposed framework is different from [12], [13], [14], and [15] where the whole training process occurs inside the enclave for a single deep-learning model. On the other hand, our framework uses a federated learning setup, and only the aggregation is performed inside the enclave. We also examine the reduction of our model's accuracy when we leverage TEE. Our proposed method has only up to 3% accuracy reduction compared to Myelin [14], and Chiron [15] that have more than 5% and 7% accuracy reduction, respectively. In other words, our proposed framework achieves a good balance between privacy and model performance.

VI. Conclusion

In this paper, a blockchain and Trusted Execution Environment (TEE) enabled Federated Learning (FL) framework is proposed for IoT. The main objective of this framework is to ensure the trustworthy aggregation of local models to obtain a global model. The aggregation is done within the blockchain network. The proposed framework leverages the Intel Software Guard Extension (SGX)-based Trusted Execution Environment (TEE) to ensure secure aggregation where each blockchain node executes the aggregation task. In this framework, each blockchain node is equipped with an SGX-enabled processor that securely generates a global model to ensure trustworthiness. Later, the global model is verified by the blockchain network via a consensus mechanism before it is added to the blockchain, thereby maintaining tamperproof storage. Users of the global model can access it and verify its integrity only through the blockchain network. We use different Convolutional Neural Network (CNN) based algorithms with several benchmark datasets to generate local models and aggregate them under FL settings. We conducted several experiments that show that our proposed framework's processing time is almost similar to that of the original FL model. In addition, our framework has only around 2% less accuracy compared to the original FL model. It is essential to mention that this framework has leveraged a hash-based consensus mechanism to ensure the model's integrity. In the future, we intend to develop an efficient consensus mechanism for the proposed TEE and blockchain-based FL framework in order to make it more practical. In this paper, we assume that all participants perform homogeneous tasks and use same approach to generate their respective local models. Each participant uses their own private dataset and the federated learning architecture to obtain a global model. However, we plan to extend our current work in the future to support heterogeneous tasks in Blockchain-based federated learning with TEE based secure aggregation.

References

[1] M. Alazab, S. P. RM, M. Parimala, P. Reddy, T. R. Gadekallu, and Q.-V. Pham, "Federated Learning for Cybersecurity: Concepts, Challenges and Future Directions," *IEEE Transactions on Industrial Informatics*, 2021.

[2] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting Unintended Feature Feakage in Collaborative Learning," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 691–706.

[3] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated Learning: Challenges, Methods, and Future Directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[4] L. Yin, J. Feng, H. Xun, Z. Sun, and X. Cheng, "A Privacy-preserving Federated Learning for Multiparty Data Sharing in Social IoTs," *IEEE Transactions on Network Science and Engineering*, 2021.

[5] Y. Liu, J. Nie, X. Li, S. H. Ahmed, W. Y. B. Lim, and C. Miao, "Federated Learning in the Sky: Aerial-Ground Air Quality Sensing Framework With UAV Swarms," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9827–9837, 2021.

[6] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated Learning with Differential Privacy: Algorithms and Performance Analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[7] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, and K.-Y. Lam, "Local Differential Privacy-Based Federated Learning for Internet of Things," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8836–8853, 2021.

[8] B. Zhao, K. Fan, K. Yang, Z. Wang, H. Li, and Y. Yang, "Anonymous and Privacy-preserving Federated Learning with Industrial Big Data," *IEEE Transactions on Industrial Informatics*, 2021.

[9] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu, and X. Zheng, "Privacy-Preserving Federated Learning Framework Based on Chained Secure Multiparty Computing," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6178–6186, 2021.

[10] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to Backdoor Federated Learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2938–2948.

[11] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa, "Oblivious Multi-party Machine Learning on Trusted Processors," in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, 2016, pp. 619–636.

[12] F. Tramer and D. Boneh, "Slalom: Fast, Verifiable and Private Execution of Neural Networks in Trusted Hardware," *arXiv preprint arXiv:1806.03287*, 2018.

[13] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "{GAZELLE}: A Low Latency Framework for Secure Neural Network Inference," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 1651–1669.

[14] N. Hynes, R. Cheng, and D. Song, "Efficient Deep Learning on Multisource Private Data," *arXiv preprint arXiv:1807.06689*, 2018.

[15] T. Hunt, C. Song, R. Shokri, V. Shmatikov, and E. Witchel, "Chiron: Privacy-preserving Machine Learning as a Service," *arXiv preprint arXiv:1803.05961*, 2018.

[16] A. Ali, H. A. Rahim, M. F. Pasha, R. Dowsley, M. Masud, J. Ali, and M. Baz, "Security, Privacy, and Reliability in Digital Healthcare Systems Using Blockchain," *Electronics*, vol. 10, no. 16, p. 2034, 2021.

[17] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-Preserving Blockchain-Based Federated Learning for IoT Devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2021.

[18] Y. Qu, S. R. Pokhrel, S. Garg, L. Gao, and Y. Xiang, "A Blockchained Federated Learning Framework for Cognitive Computing in Industry 4.0 Networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2964–2973, 2021.

[19] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-Efficient Federated Learning and Permissioned Blockchain for Digital Twin Edge Networks," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2276–2288, 2021.

[20] L. Feng, Y. Zhao, S. Guo, X. Qiu, W. Li, and P. Yu, "Blockchain-based Asynchronous Federated Learning for Internet of Things," *IEEE Transactions on Computers*, 2021.

[21] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained On-Device Federated Learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[22] R. Kumar, A. A. Khan, J. Kumar, A. Zakria, N. A. Golilarz, S. Zhang, Y. Ting, C. Zheng, and W. Wang, "Blockchain-federated-learning and Deep Learning Models for COVID-19 Detection Using CT Imaging," *IEEE Sensors Journal*, 2021.

[23] O. Samuel, A. B. Omojo, A. M. Onuja, Y. Sunday, P. Tiwari, D. Gupta, G. Hafeez, A. S. Yahaya, O. J. Fatoba, and S. Shamshirband, "IoMT: A COVID-19 Healthcare System driven by Federated Learning and Blockchain," *IEEE Journal of Biomedical and Health Informatics*, pp. 1–1, 2022.

[24] S. H. Alsamhi, F. A. Almalki, F. Afghah, A. Hawbani, A. V. Shvetsov, B. Lee, and H. Song, "Drones' Edge Intelligence over Smart Environments in B5G: Blockchain and Federated Learning Synergy," *IEEE Transactions on Green Communications and Networking*, pp. 1–1, 2021.

[25] S. Otoum, I. A. Ridhawi, and H. Mouftah, "A Federated Learning and Blockchain-enabled Sustainable Energy-Trade at the Edge: A Framework for Industry 4.0," *IEEE Internet of Things Journal*, pp. 1–1, 2022.

[26] Y. Qu, L. Gao, T. H. Luan, Y. Xiang, S. Yu, B. Li, and G. Zheng, "Decentralized Privacy using Blockchain-enabled Federated Learning in Fog Computing," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5171–5183, 2020.

[27] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient Learning of Deep Networks from Decentralized Data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based Learning Applied to Document Recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[30] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-scale Image Recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[31] V. Costan and S. Devadas, "Intel SGX Explained." *IACR Cryptol. ePrint Arch.*, vol. 2016, no. 86, pp. 1–118, 2016.

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An Imperative Style, High-performance Deep Learning Library," *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.

[33] A. Ziller, A. Trask, A. Lopardo, B. Szymkow, B. Wagner, E. Bluemke, J.-M. Nounahon, J. Passerat-Palmbach, K. Prakash, N. Rose *et al.*, "PySyft: A Library for Easy Federated Learning," in *Federated Learning Systems*. Springer, 2021, pp. 111–139.

[34] C.-C. Tsai, D. E. Porter, and M. Vij, "Graphene-sgx: A Practical Library {OS} for Unmodified Applications on {SGX}," in *2017 {USENIX} Annual Technical Conference ({USENIX}{ATC} 17)*, 2017, pp. 645–658.

[35] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[36] A. Krizhevsky and G. Hinton, "Convolutional Deep Belief Networks on CIFAR-10," *Unpublished manuscript*, vol. 40, no. 7, pp. 1–9, 2010.

[37] L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[38] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-resnet and the Impact of Residual Connections on Learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[39] M. Fredrikson, S. Jha, and T. Ristenpart, "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1322–1333.

[40] B. Wang and N. Z. Gong, "Stealing Hyperparameters in Machine Learning," in *2018 IEEE Symposium on Security and Privacy (SP)*, 2018, pp. 36–52.

[41] M. S. Rahman, I. Khalil, and M. Atiquzzaman, "Blockchain-powered policy enforcement for ensuring flight compliance in drone-based service systems," *IEEE Network*, vol. 35, no. 1, pp. 116–123, 2021.

[42] S. H. Alsamhi, O. Ma, M. S. Ansari, and F. A. Almalki, "Survey on Collaborative Smart Drones and Internet of Things for Improving Smartness of Smart Cities," *IEEE Access*, vol. 7, pp. 128 125–128 152, 2019.

[43] M. S. Rahman, I. Khalil, and M. Atiquzzaman, "Blockchain-Enabled SLA Compliance for Crowdsourced Edge-Based Network Function Virtualization," *IEEE Network*, vol. 35, no. 5, pp. 58–65, 2021.