electronics

MDPI

Article

An End-Process Blockchain-Based Secure Aggregation Mechanism Using Federated Machine Learning

Washington Enyinna Mbonu, Carsten Maple * and Gregory Epiphaniou *

WMG, University of Warwick, Coventry CV4 7AL, UK; washington.mbonu@warwick.ac.uk
* Correspondence: cm@warwick.ac.uk (C.M.); gregory.epiphaniou@warwick.ac.uk (G.E.)

Abstract: Federated Learning (FL) is a distributed Deep Learning (DL) technique that creates a global model through the local training of multiple edge devices. It uses a central server for model communication and the aggregation of post-trained models. The central server orchestrates the training process by sending each participating device an initial or pre-trained model for training. To achieve the learning objective, focused updates from edge devices are sent back to the central server for aggregation. While such an architecture and information flows can support the preservation of the privacy of participating device data, the strong dependence on the central server is a significant drawback of this framework. Having a central server could potentially lead to a single point of failure. Further, a malicious server may be able to successfully reconstruct the original data, which could impact on trust, transparency, fairness, privacy, and security. Decentralizing the FL process can successfully address these issues. Integrating a decentralized protocol such as Blockchain technology into Federated Learning techniques will help to address these issues and ensure secure aggregation. This paper proposes a Blockchain-based secure aggregation strategy for FL. Blockchain is implemented as a channel of communication between the central server and edge devices. It provides a mechanism of masking device local data for secure aggregation to prevent compromise and reconstruction of the training data by a malicious server. It enhances the scalability of the system, eliminates the threat of a single point of failure of the central server, reduces vulnerability in the system, ensures security, and transparent communication. Furthermore, our framework utilizes a fault-tolerant server to assist in handling dropouts and stragglers which can occur **in** federated environments. To reduce the training time, we synchronously implemented a callback or end-process mechanism once sufficient post-trained models have been returned for aggregation (threshold accuracy achieved). This mechanism resynchronizes clients with a stale and outdated model, minimizes the wastage of resources, and increases the rate of convergence of the global model.

Keywords: artificial intelligence; deep learning; federated learning; blockchain; secure aggregation

Citation: Mbonu, W.E.; Maple, C.; Epiphaniou, G. An End-Process Blockchain-Based Secure Aggregation Mechanism Using Federated Machine Learning. *Electronics* **2023**, *12*, 4543. https://doi.org/10.3390/electronics12214543

Academic Editors: Mikolaj Karpinski, Oleksandr O. Kuznetsov and Roman Oliynykov

Received: 19 September 2023
Revised: 31 October 2023
Accepted: 1 November 2023
Published: 5 November 2023

1. Introduction

The introduction of the Internet of Things (IoT) has resulted in the massive growth in the number of intelligent devices. With strong hardware and dedicated sensors, these devices can collect and process data at high speed. Artificial Intelligence (AI) and Machine Learning (ML) flourish in data. These data are generated by billions of IoT devices and smart phones. By generating these large amounts of data, the IoT has effectively enhanced the training of Deep Learning (DL) models. However, IoT devices cannot independently execute DL algorithms because of their resource-constrained nature. Traditionally, a DL approach entails data collection from various sources and storing them in a centralized location. These stored data are then used to train the DL model. However, privacy legislations such as European Commission's General Data Protection Right (GDPR) and the U.S. Consumer Privacy Bill of Right require that in certain cases, data collection may not be feasible. To address this issue, Federated Learning (FL) was introduced. FL is a distributed DL technique that creates a global model through the local training of multiple

decentralized edge devices. It enables distributed ML to be effectively accomplished between various edge devices or participants. Moreso, it promotes the exchange of big data and tends to enhance the privacy preservation of users' data within the confinement of the law [1,2].

The FL algorithm permits the decentralized training of data, but the central server aggregates the model and process planning. In traditional FL, the central server sends to each participating device/client an initial/pre-trained model for training. Using their own local dataset, each participating device trains the model locally and sends it back to the central server for aggregation. The server aggregates the returned trained model to produce an updated global model that is sent back to the participating devices for another round of local training [3]. This client–server interaction [4] continues until model convergence is achieved or a specific number of iterations (rounds) are attained. However, this centralized approach of model aggregation and process planning in traditional FL makes the central server a single point of failure [5]. This threat of a single point of failure (SPOF) on the server could be because of unforeseen external attacks, purposeful unfair aggregation, unexpected failure in network connection, etc. This strong dependence on the central server is a significant drawback to this technique because if there exist a problem with the server or it fails, the training process will stop and as mentioned earlier, the resource-constrained end devices will not be able to independently execute the aggregation process [6]. Several risks and issues arise in such a centralized model: (1) *Communication failure*: To collect model updates and distribute the updated model, the central server depends on communication with end devices. If there is a communication failure, it can interrupt the training process and delay model updates. (2) *Scalability and overload issues*: The central server might face scalability issues in a large network with several end devices. If the model updates and requests from end devices cannot be effectively handled by the central server, it may be overloaded and slow down or crash. This will lead to training disruption. (3) *Security breach*: A security breach on the central server could result in malicious actors gaining unauthorized access to sensitive data or model updates, leading to privacy issues or tampering with the model updates. (4) *Server downtime*: The central server may experience hardware failures or software issues which could result in downtime, making it unavailable to end devices. During this period, model updates cannot be aggregated, and the FL process will stop. (5) *Aggregation bias*: To form an updated global model, the central server aggregates updates from various end devices. If the aggregation is biased, it could favour certain end devices over others, leading to a skewed model result.

Furthermore, the privacy leakage in FL could put updates from the end devices at risk due to fairness and trust issues from the central server, and this could be because of the following: (1) *Central server integrity*: The central server orchestrates the training and aggregation of model updates from end devices. If the server is compromised, it could change or alter the model updates, resulting in influenced or poisonous models being dispersed to end devices. (2) *Model poisoning*: Without thorough validation, the central server may aggregate model updates from a malicious participant in the training process. The malicious participant may attempt to poison the global model by intentionally sending updates that degrade the model performance. (3) *Data bias*: Data distribution across end devices may not be evenly distributed, resulting in bias or data imbalance. This imbalanced distribution could result in less accurate models and be unfair to a subset of the end devices. (4) *Data privacy and security*: In as much as FL aims to preserve the privacy of the user data by not sharing raw data with the central server, there is still risk of data exposure during model updates. The gradients sent to the server may accidentally reveal sensitive information about the local data. Moreso, a malicious central server might compromise or gain sensitive intuitions of the updates from the end devices because of its capability to successfully reconstruct the original data due to non-scrutinized, constant, and direct communication with the end devices. Recent works have shown that a malicious server can use the gradient information to infer the sensitive content about the clients' training data. Through a Generative Adversarial Network (GAN), the distribution of the training

data can be recovered by the malicious server [7]. Also, attacks on the server can alter the global model [8]. Furthermore, attacks on the end devices could manipulate local models, and this can result in errors in the global model generated from such altered local models. In like manner, the integrity of the generated global model should be verified before use by the edge devices. FL was integrated with Blockchain technology to ensure transparency and enhance its privacy preservation, security, and performance [9,10].

To address this SPOF threat, privacy, trust, fairness, transparency, and security, Blockchain is integrated into FL methodology to mitigate against vulnerability in the FL centralized approach of model aggregation and process planning. Blockchain is used as a reliable orchestrating memory that eliminates the need for a central coordinating unit and provides a secured, certified, and validated exchange of information. The three fundamental security considerations identified in Ref. [11] are confidentiality, integrity, and availability. As identified in Refs. [12,13], FL suffers from insufficient incentives, poisoning attacks, privacy preservation, etc.

In Blockchain, transactions are unaltered and timestamped. As a distribute ledger, Blockchain can act as an append-only database that offers data integrity. Also, it can act as a hybrid Blockchain that guarantees data confidentiality to only authenticated and permitted users. Blockchain allows the storage and exchange of data in a decentralized approach using digital blocks, increasing FL fault tolerance capacity [14]. These digital blocks are chained together using cryptographic hashes to form a distributed ledger. Blockchain is a type of distributed ledger that is shared among all devices in a federated network. This ensures that data are immutable, visible, traceable, transparent, and non-repudiated. These unique characteristics of Blockchain make it an ideal technology to combine with FL to safeguard the privacy and security of aggregated data.

This paper aims to implement a callback/end-process Blockchain-based secure aggregation mechanism for FL through the masking of model updates. For each iteration of the FL training process by the central server, the Blockchain enables the masking and tracking of local models, where devices mask their local model to train the global model, and post-trained models are sent back to the server for model aggregation. When a certain percentage of post-trained models have been returned by the clients, the server will implement a callback aggregation and issue a force stop to lessen training time, reduce communication rounds, and speed up the convergence of the global model. Similarly, if this percentage as stated by the FL server is not met and a deadline has been reached, an end-process strategy will be issued to the clients yet to return their post-trained models to avoid the issue of infinite loop or endless waiting.

In both cases (callback/end-process), devices at the stage of model training will be forced to synchronize with the central server. The main contributions of this paper are as follows:

- Formulate a mechanism of masking the local models to train the global model for aggregation by the server to prevent the compromise and reconstruction of the data used to train the model.
- Implement a Blockchain network for transparent communication within an FL environment which eliminates the threat of SPOF, ensures transparency, and enhances the security and privacy preservation of data.
- To lessen training time due to dropouts that might occur in the FL environment, a callback function will be synchronously implemented by the FL server once sufficient post-trained models have been returned.

The rest of the paper is organized as follows: In Section 2, we examine related works in the field. Section 3 is the background information of Blockchain technology. Section 4 illustrates the distribution of the global model, consensus mechanism, masking of device local data, and model aggregation. Section 5 introduces the system architecture, synchronization process, client selection update, and FL loss function. Section 6 illustrates the callback function and end-process aggregation mechanism in FL. Section 7 presents the Results, Discussion, and performance evaluation. Finally, Conclusions are drawn in Section 8.

2. Related Work

2.1. Secure Aggregation in FL

To guarantee privacy and security using FL, the following proposals [15,16] on secure aggregation mechanisms have been proposed. Fereidooni et al. [15] proposed a secure aggregation for private Federated Learning. This approach tends to impede inference attacks on FL by prohibiting access and tampering with trained model updates. They utilized a Secure Multipath Computation (SMC) encryption technique to prevent the aggregator from accessing the model updates used for the training of the Machine Learning model. Similarly, Wu et al. [16] proposed a secure aggregation mechanism for model updates in FL to prevent inference and inversion attacks that can obtain sensitive information from local model updates. Their approach utilized matrix transformation to protect each clients' model updates by preventing the attacker from gaining sensitive information using encryption of a little part of the model update to avoid heavy encryption that could result in low accuracy. Their aggregation mechanism functions with an acceptable overhead. However, both approaches suffer the threat of the SPOF of the central server which orchestrates the training process [5].

Huang et al. [7] proposed a secure aggregation mechanism for Federated Learning that utilized ransom masking code to ensure the confidentiality of local gradients. Their proposed mechanism ensures the confidentiality of local gradients and verifiability of aggregated gradients. However, this mechanism is not communication- and bandwidth-efficient when several clients are involved in the training process. Also, it suffers from the threat of SPOF in the aggregator and verification servers. To protect against Byzantine adversarial that could compromise the performance and convergence of the global model, Zhao et al. [17] proposed a secure aggregation mechanism in FL. This mechanism used intel SGX primitives to ensure privacy preservation of the local models by providing a recovery key to the encrypted models. This technique ensures that sensitive information is not revealed to the aggregation server. However, it still suffers the threat of SPOF of the aggregation server that could halt the training process.

2.2. Blockchain-Based Federated Learning

Traditional FL mechanisms depend on the central server for coordination and orchestration. This central server dependence may result in SPOF, trust issues, and unwanted behaviours of the server. To ensure effective decentralization, trust, transparency, and reliability, Blockchain technology has emerged. Blockchain technology has been implemented by many researchers to eliminate the threat of SPOF in traditional FL [18,19].

To guarantee data authenticity and privacy protection, the authors in Ref. [18] implemented an FL framework using Blockchain in self-driving cars. In Ref. [20], they implemented a private Blockchain FL using an interstellar file system to minimize high storage costs in Blockchain, inference, and poisoning attacks in FL. As seen in Ref. [21], they implemented a private Blockchain for secure model aggregation in FL using a consensus process for traffic prediction. In Ref. [19], the author proposed a Blockchain-enabled FL where the security and privacy of the user's information were protected by encrypting and encoding it in the cloud. All these research works mentioned above makes use of Blockchain technology for the aggregation of a trained model, which incurs huge bandwidth and complexity in computation. Most of the contributions are based on a private Blockchain, where the entire process is not decentralized, which could result in trust issues.

For the local evaluation and global aggregation of parameters, Sun et al. [22] proposed the use of Blockchain in FL to lessen the effect of end-point adversarial training data. In this work, the method of selecting a committee member is not feasible and was not fully analysed. Furthermore, if there are more users participating in the network, the method may experience a decrease in classification accuracy. To facilitate the model update and guarantee secure aggregation of the global model, Mallah et al. [23] proposed a Blockchain-enabled Federated Learning that selects only reliable IoT devices for global aggregation. Their approach ensures the aggregation of the global model through optimized behaviour

monitoring of the devices, increasing the convergence time of FL processes while preserving network performance. However, there is a trade-off in time and bandwidth efficiency, and the scalability of this technique in variable network topology is not guaranteed. To guarantee a secure aggregation mechanism that will ensure trust, security, and integrity of the global model, the following approaches [24,25] have been proposed.

Kalapaaking et al. [24] proposed a Blockchain-based FL secure aggregation mechanism to guarantee the security and integrity of the global model. Their technique ensured a trusted aggregation of the local model to generate a global model. However, they failed to consider how to handle stragglers and dropouts in Industrial IoT (IIoT). Their assumption was that all the IIoT will successfully return their trained model, which is practically impossible. Chen et al. [25] proposed a Blockchain-based FL for the secure aggregation and efficient sharing of medical data. Their technique enhanced the sharing of medical data in a privacy-preserved manner. However, the use of a contribution-weighted aggregation mechanism, as seen in Ref. [25], will incur huge bandwidth and complexity in computation, which makes the technique not feasible within a resource-constrained setting. To minimize the impact of the attacks from malicious clients or a poisonous server and preserve privacy in FL, Refs. [26,27] have been proposed.

Li et al. [26] proposed a Blockchain-based decentralized FL with committee consensus to solve the issues of SPOF, privacy, and security. Their technique solves the threat of SPOF, prevents malicious attacks, prevents models from been exposed to poisoning or unauthorized devices, and the burden of consensus computing is reduced. However, the validation consumption is increased, and the consensus committee selection could result in security issues if not properly selected. Miao et al. [27] proposed an FL privacy preserving scheme based on a Blockchain network. Their approach mitigates against a poisoning attack from malicious clients and ensures a transparent process using the Blockchain network. However, they did not provide mechanisms on how to deal with stragglers and dropouts that may exist within the devices.

In comparison with above-mentioned research works, our approach is scalable and eliminates the threat of SPOF with traditional FL while retaining the central server for the aggregation of post-trained models. This significantly reduces the computational burden and communication cost of aggregating the trained model using the Blockchain network. Additionally, our technique handles the issues associated with stragglers and dropouts in IoT systems within an FL environment.

3. Background Information on Blockchain Technology

Distributed Ledger Technology (DLT) is an umbrella technology of Blockchain in the sense that every Blockchain is a DLT but not every DLT is a Blockchain. Previously, Blockchain was primarily designed for digital transactions and used as currency, but recently, researchers have found various ways of using the technology or combining it with other methodologies for the greater good. In a distributed ledger, data are independently held and updated by end devices within the network. This eliminates the need for a central authority to perform the orchestration. Rather, each end device is given access to the transaction lists, where each individually and autonomously updates the distributed ledger. Implementing Blockchain with FL will provide additional protection, strong robustness, and privacy preservation.

In Ref. [28], the Blockchain technology components were identified as the Blockchain, Blockchain network, and distributed consensus mechanism. A Blockchain network comprises two computation nodes, namely, the verifier and normal nodes. The former hold an entire record of the Blockchain structure and transaction validations, implement smart contracts, ensure data security, and require high storage and computational capability, while the latter do not keep a record of the Blockchain ledger due to limited computational and storage capability but obtain a little knowledge from the full nodes about the Blockchain status. In private and distributed data, Blockchain has proven to be a secure aggregation mechanism for edge computing in a federated Machine Learning (FML) environment.

There exist three types of Blockchain: (1) Public Blockchain: This type gives free access to the public or any person to partake in the core activities. It is a democratically decentralized Blockchain operation. The disadvantage with this type of Blockchain is that deceptive participants may exist that could execute malicious activities on core functions. (2) Private Blockchain: In this type, only chosen and validated participants are allowed to join. A control function is put on who can partake in the core activities. These types of Blockchain are not essentially decentralized because the distributed ledger is operated by central supervisors and could result in trust issues. (3) Hybrid Blockchain: this combines public and private contributors. It could involve external parties that carefully implement network restrictions and control contributor activities in their respective roles. The immutability data structure of Blockchain makes it a viable technology when implemented in an FL framework. To deal with operational changes or issues such as stragglers or dropouts in an FL environment, Blockchain offers an efficient ecosystem in handling such issues. In a bid to identify malicious activities within a data auditing scheme, Ref. [29] described Blockchain as a DLT that keeps track of the activities of all the nodes in a Blockchain network using a smart contract. The advancement of Blockchain technology has aided the implementation of smart contract technology. Smart contracts are treaties between various entities based on a particular matter that is meant to be implemented by computer programs. Using smart contracts, more users are encouraged to participate in FL training, facilitating the management and control of the entire process [30]. Blockchain members validate and verify codes in the form of smart contracts to protect relationships over computer networks. Smart contracts are used by Blockchain members to make a treaty in a distribution ledger without involving a central third party to implement the treaty. Using interaction records between nodes on a Blockchain network, smart contracts can effectively and automatically identify violations based on the records [31]. To ensure the correctness of the smart contracts, nodes within the Blockchain network must run the same smart contracts, and through a consensus agreement strategy, results are accepted. With Blockchain and smart contracts, various fields have been expanded and improved [32].

3.1. Structure of Blockchain

There are five logical layers of Blockchain, and they are as follows:

- *Application layer*: In Blockchain structure, this is the uppermost layer. The application layer serves as a channel for the Blockchain to connect to the real world. It comprises the chain code, smart contracts, and distributed applications (dApps). The two sub-layers of the application layer are the presentation and execution layers.
- *Consensus layer*: In this layer, consensus algorithms are used to validate transactions. A method of agreement must be reached to generate a new block on a single data block comprising multiple insecure numbers of nodes. These methods of agreement are termed consensus algorithms. They are used to validate transactions and to determine the node to generate a new block. The consensus mechanisms are Proof of Work (PoW), Proof of Stake (PoS), Proof of Elapsed Time (PoET), Proof of Authority (PoA), and Byzantine Fault Tolerance (BFT). We adopted PoA as a consensus mechanism for this framework.
- *Network layer*: In a Blockchain network, the network layer provides communication between nodes. This is also referred to as a Point-to-Point (P2P) network. Network failures are avoided in P2P networks because the nodes regularly communicate with each other. P2P networks help filter out illegal transactions. Full nodes and light nodes are the two types of nodes in P2P networks.
- *Data layer*: This is the basic layer of the Blockchain structure. The data layer comprises data blocks, a digital signature, transactions, a Merkle tree, and hash functions.
- *Infrastructure layer*: This layer is also known as the hardware structure. It contains the services that enable data exchange. Within the infrastructure layer exist services, virtual machines, messaging, and containers.

3.2. Performance Evaluation of Blockchain

The performance of Blockchain can be evaluated with the following factors:

- *Decentralization*: The decentralization nature of Blockchain effectively eliminates SPOF and solves the bottleneck problem of a central authority. The operation of a Blockchain network is unaltered by the disruption of a single node in the network because data exist in multiple nodes on the P2P network. This p2p network configuration ensures the immutability and authenticity of data. The decentralization nature of a Blockchain network effectively handles dropouts or offline nodes without compromising the security and availability of the network.
- *Transparency*: There is great transparency in Blockchain transaction histories because nodes in the network share the same documents. The shared documents must be modified through a consensus where every node in the network must agree. Any alteration of a single record will require the modification of subsequent records for the entire network. With transparency, the integrity of the network is protected, and there is a complete reduction in data alteration.
- *Improved security*: There is enhanced security in Blockchain technology implementation because an agreement must be reached in advance before a transaction takes place. After a transaction is approved, it becomes encrypted and connected to the previous one. To avoid any potential security breaches, the data are not stored on a single server but are instead distributed across a network of computers. Private/public key infrastructures are used to improve the security of the Blockchain network, and it is mathematically impossible to devise these keys because they are randomly generated strings and numbers. Through this process, the security of the network is strengthened, and there is a significant reduction in data leakage.
- *Immutability*: Using cryptographic hashes and timestamps, Blockchain ensures that data remain immutable. After validation, the hash function restricts data altering, updating, and removal. Any change in the transaction data can be identified easily.
- *Data privacy*: In a Blockchain network, data are protected against alteration using digital signatures. Immutable hash chains ensure that transactions are monitored by nodes across the network to preserve data rights.
- *Anonymity*: The data on the chain in a Blockchain network are public. However, Blockchain uses encryption techniques to achieve the privacy preservation of the end devices' private data to avoid exposure to another node on the network.

3.3. Technical Limitations

The technical limitations of Blockchain include the following:

- *Computational complexity*: There is a high computational cost involved in completing a transaction. It entails several steps, such as validation, scrutiny, and security checks across multiple nodes. This computational complexity consumes a significant amount of power and resource-constrained IoT devices will need help to meet resource demands. Also, the sophisticated architecture will demand high computational capabilities that could result in an increase in implementation and running costs.
- *Privacy and security issues*: Blockchain can resist security attacks such as Distributed Denial-of-Service (DDoS), Ransomware, and Sybil attacks. However, there are integral security shortcomings in existing Blockchain networks. If the computing resources can be controlled by more than half of the nodes running Blockchain, consensus processes could be altered for malicious reasons. This is known as a 51% attack. Furthermore, if transactions are not robustly supervised, Blockchain could suffer network interruption and data loss.
- *Scalability issues*: The limited scalability of Blockchain is caused by restricted throughput and high computational costs. This negatively impacts the overall system performance due to the limited block size and increased block time. These complications arise when processing large amounts of data on the Blockchain, especially in large IoT systems where massive amounts of data are generated [33].

According to Ref. [34], Blockchain is defined as a decentralized and distributed technology that can be used and employed in applications involving daily living, such as healthcare systems, supply chain management, digital currencies, etc.

3.4. Blockchain Technology Applications

The applications of Blockchain technology include the following:

- *Secured digital payment system*: Blockchain technology ensures a secured digital payment system and facilitates a reduction in intermediaries' fees as compared to traditional digital payments, where organizations such as credit card companies and financial institutions act as intermediaries. Also, the transaction time is drastically reduced using automated validation and verification systems [35].
- *Automated governance*: Blockchain uses E-governance to provide automated government services. These services include tax collection, conducting elections, issuing certificates, implementing social security, etc. These services are enhanced, and personal data privacy is preserved using Blockchain technology. It gives adequate control functions and supports the efficient management of these services.
- *Data redundancy*: Blockchain facilitates efficient data distribution. Distributed data storage is one of the features of Blockchain and it helps to easily spot data alteration and facilitates recovery from peer nodes. This attribute helps to keep good audit records of data and ensures data integrity and confidentiality.
- *Supply chain management*: This involves business processes that go through various steps to supply the needs of customers and add value to stakeholders. It involves the synchronization of complex processes that require efficient monitoring and accountability.

Having examined the analysis of the Blockchain technology, the next section will showcase global model distribution and the performance evaluation of the Blockchain technology through the masking of device local data for privacy preservation and mitigation against the reconstruction of device local data by a malicious server.

4. Global Model Distribution, Consensus Mechanism, Masking of Device Local Data, and Model Aggregation

4.1. Global Model Distribution

The aggregator server (central server) initiates the training process by sending an initial or pre-trained model to the clients through the Blockchain network. The Blockchain network, using the nodes, verifies the model, validates it, and reaches a consensus.

4.2. Consensus Mechanism

We implemented a private Blockchain setting and adopted PoA as a consensus mechanism. Using trustworthy validators, transactions and blocks are validated. These validators are also tasked with the responsibility of creating new blocks and transaction confirmation. In a consensus mechanism using PoA, not all the nodes are allowed to participate in the consensus process; rather, validators are chosen based on attributes such as investment in the system, identity verification, and reputation. Based on these attributes, PoA depends on a pre-selected group of nodes as validators. Unlike PoW, where there is competition to solve a puzzle, there exist no competition to create a block among validators in PoA. Rather, validators take turns based on a set schedule or a round-robin fashion.

In our architectural framework, the global model sent from the aggregator server to the clients is checked and validated and a consensus is reached before it is sent to the clients as a smart contract. Using a pre-selected group of nodes as validators and turn taking based on a set schedule or a round-robin fashion, the speed of the transaction verification process is significantly increased. Once transactions are confirmed and validated by validators, they are added to the next block. This process ensures trust and security because validators have a strong incentive to correctly validate transactions, or their reputation will be at stake. However, through governing processes, malicious validators can be removed from the network by other validators.

Regarding speed and scalability, transactions are processed faster in PoA networks than in PoW and PoS because PoA does not require stake-based competitions and computations that are resource-intensive. In energy efficiency, PoA is more environmentally friendly because it does not depend on resource-intensive mining. Furthermore, block creation and transaction validation are more predictable with a set schedule and known validators. There is reduced centralization risk in PoA. In an FL setting like our architectural framework, where trust among clients can be established and maintained, PoA offers a balance between efficiency and decentralization.

4.3. Masking of Device Local Data

The essence of masking the local model is the privacy preservation of a device's local data and mitigation against the server from reconstructing the data used to train the model. To illustrate the masking of the device local model using common seeds, keeping each device local model private and secure using a Blockchain network, the following assumptions are made:

Assume M_{12}, M_{13}, *and* M_{14} as the masks (M) indiscriminately created based on common seeds by (*device*1 and *device*2), (*device*1 and *device*3), and (*device*1 and *device*4), respectively. This is such that ($M_{12} = M_{21}$), ($M_{13} = M_{31}$), and ($M_{14} = M_{41}$), respectively. Similarly, (M_{23}, M_{24}, *and* M_{34}) are assumed to be the masks indiscriminately created based on common seeds by (*device*2 and *device*3), (*device*2 and *device*4), and (*device*3 and *device*4), respectively. In like manner, ($M_{23} = M_{32}$), ($M_{24} = M_{42}$), and ($M_{34} = M_{43}$), respectively. Using a key agreement protocol, these common seeds are decided prior to training amongst a pair of devices. The following equations ensued to further illustrate the individual device and the secure aggregation protocol.

Let device = D.

Global model = θ_G.

Device local model = $\beta_1, \beta_2, \beta_3, \ldots, \beta_n$.

Training samples/dataset = $x_1, x_2, x_3, \ldots, x_n$.

$$D_1 = \beta_1 x_1 + M_{12} + M_{13} + M_{14} \tag{1}$$

$$D_2 = \beta_2 x_2 - M_{21} + M_{23} + M_{24} \tag{2}$$

$$D_3 = \beta_3 x_3 - M_{31} - M_{32} + M_{34} \tag{3}$$

$$D_4 = \beta_4 x_4 - M_{41} - M_{42} - M_{43} \tag{4}$$

Combining (1) to (4) will result in the mask cancelling out, i.e., ($M_{12} - M_{21} = 0$), ($M_{13} - M_{31} = 0$), ($M_{14} - M_{41} = 0$), ($M_{23} - M_{32} = 0$), ($M_{24} - M_{42} = 0$), *and* ($M_{34} - M_{43} = 0$).

$$\sum D_i = D_1 + D_2 + D_3 + D_4 = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 \tag{5}$$

$$\theta_G = \sum \frac{D_i}{\sum x_i} = \sum \frac{(\beta_i x_i)}{\sum x_i (i > 3)} \tag{6}$$

In summary, the local model masked at device i is as follows:

$$Di = \beta_i x_i + \sum_{1<2} M_{12} - \sum_{1>2} M_{21} \tag{7}$$

4.4. Aggregation of Masked Trained Model

Figure 1 illustrates how the device masked trained models are aggregated at the server. As a secured end-process Blockchain-based aggregation mechanism, only returned post-trained models are needed for aggregation. An end-process or a callback function mechanism forces the devices to synchronize with the server. During aggregation, the

callback or end-process mechanism eliminates the problem of complicated handling of stragglers and dropouts masked using double masking and Shamir's t-out-of-n Secret Sharing [21]. This mechanism ensures that only returned post-trained models are needed for aggregation.

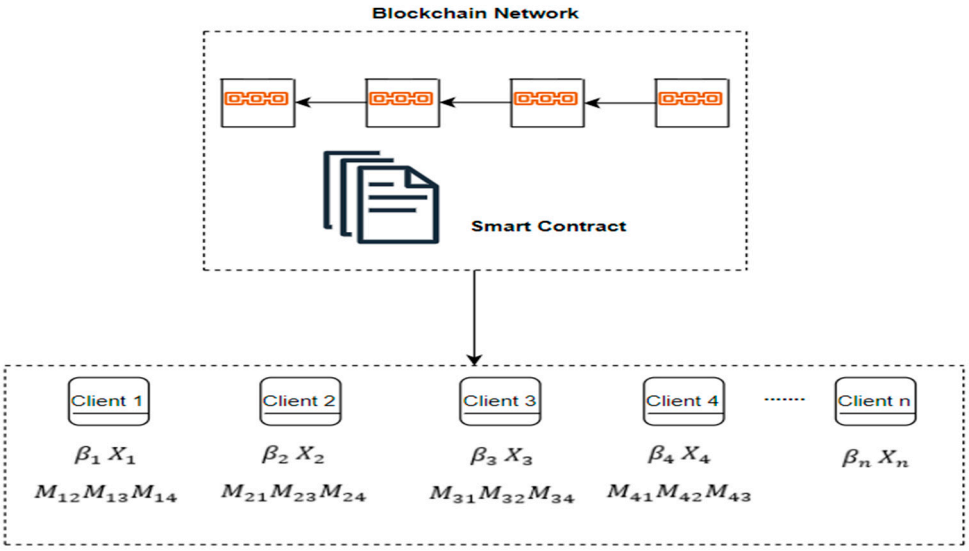

Figure 1. Example of masking of the global model using the Blockchain.

The aggregated global model is as follows:

$$\theta_G = \frac{(\beta_1.\, x_1 + \beta_2.\, x_2 + \beta_3.x_3 + \cdots \beta_n.x_n)}{(x_1 + x_2 + x_3 + \cdots x_n)} \tag{8}$$

$$\theta_G = \frac{\sum (\beta_i x_i)}{\sum x_i} \tag{9}$$

where $i = 1,\ 2,\ 3,\ \ldots, n$.

As shown in Figure 2, the masked models will be sent to the server for aggregation. The server periodically sends the intermediate results of the aggregated model to the Blockchain network, and the Blockchain, through the fault-tolerant server, sends an encrypted update of the clients' situation to the aggregator server. This process will be fully illustrated in the next section. The local model trained at device i (β_i) using its local dataset (x_i) trains the model, and the updates are sent to the aggregator server for aggregation as masked model updates. The masked model updates are aggregated and unmasked to create a new global model. The new global model is sent back to the devices through the Blockchain network for further training, and this iteration continues until convergence or the desired accuracy is achieved, as shown in Figure 2.

A new global model emerges as follows:

$$\theta_G = \sum \frac{D_i}{\sum x_i} \tag{10}$$

At the server, the mask is cancelled, resulting in the following:

$$\theta_G = \frac{\sum \beta_i\, x_i}{\sum x_i} \tag{11}$$

where $i = 1,\ 2,\ 3,\ \ldots, n$.

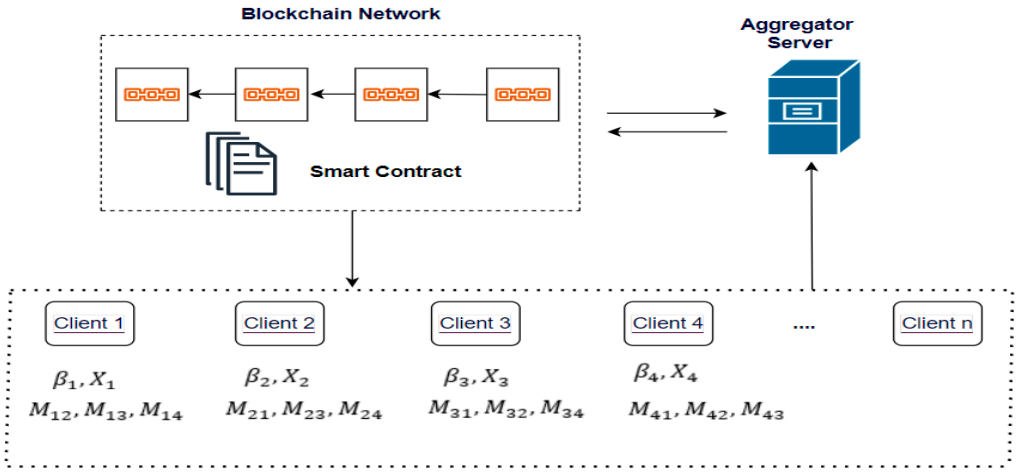

Figure 2. Local model masking and secure aggregation mechanism in FL.

5. System Architecture, Synchronization Process, Client Situation Update, and Loss Function

5.1. System Architecture

In this section, a brief description of all the entities that make up the architectural framework as illustrated in Figure 3 is given, followed by a general explanation of the system architecture, client situation updates (CSUs), and how the loss function is minimized using federated averaging (FedAvg). From Figure 3, the Blockchain-based secure aggregation system architecture consists of four entities: the *aggregator server*, *blockchain network*, *fault-tolerant server*, and *clients*. Their roles are described as follows:

- *Aggregator server*: This is a central server that is tasked with aggregating the model updates from the clients. It generates the initial global model needed for training and sends it to the clients through the Blockchain network.
- *Blockchain network*: The Blockchain network ensures trust and transparency and enables the masking of the global model for local model training. The intermediate results are sent to the Blockchain network for an efficient and transparent computation process.
- *Fault-tolerant server*: In the context of the Blockchain, this entity acts like the PoA, a variation of PoS that is less energy-intensive and requires less computing power when compared with PoW. It ensures that the failure of certain clients or clients going offline does not prevent the operation of the network. For every iteration, the clients' situation updates are communicated to the fault-tolerant server in a privacy-preserved manner. These updates are used within the network to handle the issues associated with stragglers and dropouts.
- *Clients*: They are data owners that train the global model. They send their model updates to the central server and status report (active, stragglers, and dropouts) to the fault-tolerant server in a privacy-preserved manner.

From Figure 3, an initial or pre-trained global model from the *aggregator server* is sent to the clients through the Blockchain network for model training and aggregation. The trustworthy nodes in the *blockchain network* verify the model, reach a consensus using PoA, mask the global model, and send it to the clients as smart contracts. This ensures the security, transparency, tamper-proofness, and privacy-preservation of data. Each client uses their local data to train the model and send a masked update to the central server for aggregation. The masking of device local data and secure aggregation is illustrated using Algorithm 1.

Algorithm 1: Masking Local Data and Secure Aggregation

Input: Device D, global model θ_G, device local model $\beta_1, \beta_2, \beta_3, \ldots, \beta_n$, training sample $x_1, x_2, x_3, \ldots, x_n$, masks M, number of training rounds T

Output: The aggregated final model θ_T

Initialization:
(a). The global model θ_G from the aggregator server is sent to the clients/devices for model training through the Blockchain network;
(b). The Blockchain establishes a secure and transparent channel between the aggregator server and the clients;
(c). θ_G is validated using trustworthy validators;
(d). Consensus is reached using PoA;
(e). Initialize the training round index by t = 1.
Procedure:
for t ≤ T **do**
    (I). **For Blockchain network**:
    repeat
        for each device **do**
            (a). Decide a key agreement protocol amongst pair of devices;
            (b). Mask (M) indiscriminately between device1 and device2 such that $M_{12} = M_{21}$;
            (c). Mask for all connected devices (M_{12}, M_{13}, M_{14}), (M_{21}, M_{23}, M_{24}), (M_{31}, M_{32}, M_{34}), (M_{41}, M_{42}, M_{43}) as represented in Figure 1;
            (d). Split the masked model using a secret sharing scheme across the Blockchain network;
            (e). Store model shares on the Blockchain to ensure decentralization and transparency;
            (f). Send to the devices using smart contract;
        end
    until The local mask model is as (9);
    (II). **For devices**:
    for ∀D1 ∈ Dn **do**
        (a). Devices train the model using their local data ($D_i = \beta_i x_i$);
        (b). Each device sends its trained models to the aggregator server according to (3)–(6);
    end
    (III). **For the aggregator server**:
    for θ_G **do**
        (a). Aggregate and unmask according to (8);
        (b). Dynamically initiate a callback or end-process aggregation mechanism or get update about the device status as per Algorithm 2;
        (c). Generate a new global model according to (11);
    end
    t ← t + 1.
end
return The aggregated final model θ_T.

There is constant communication between the central server and the *fault-tolerant server* through the Blockchain network. The clients communicate their status updates to the fault-tolerant server in a privacy-preserved manner. As illustrated in Algorithm 2, the fault-tolerant server handles stragglers and dropouts in an efficient manner by ensuring that the unavailability of participating clients does not stop the training process because other clients can step in and take their place. The client's status feedback is communicated to the central server through the Blockchain network in a privacy-preserved manner. The central server uses this information to decide when to initiate a callback function or end-process aggregation mechanism. These aggregation mechanisms effectively handle stale

model updates which could occur when clients experience connectivity issues or delays in sending model updates to the central server. Also, it resynchronizes clients that experience a temporary drop-out and then reconnects with an outdated model to receive the latest global model. The entire FL training process is recorded in the Blockchain network, and this technique addresses the threat of an SPOF, privacy, trust, and security. Furthermore, the aggregation of post-trained models (model updates) is carried out by the aggregator server, which greatly reduces the computing burden on the Blockchain network. Furthermore, the clients' status communications to the fault-tolerant server help to manage the issues associated with stragglers and dropouts.

Algorithm 2: Client Situation Update

Input: Device D, set of clients A, selected clients B, crash clients C, Post-trained P, selection fraction S, number of training rounds T.
Output: Crash ratio C_r
$|P| = S.|A|$
Initialization:
a). Initialize the training round index by t = 1.
Procedure:
for t ≤ T **do**
(I). For clients:
Send status update according to (12);
(II). For fault tolerant server:
(a). Compute C_r according to (13);
if $C_r > |P|$ **then**
send C_r update to the Blockchain network;
else
do nothing;
end
(III). For the Blockchain network:
(a). Update aggregator server;
(b). Add D;
end
t ← t + 1.
end
return C_r

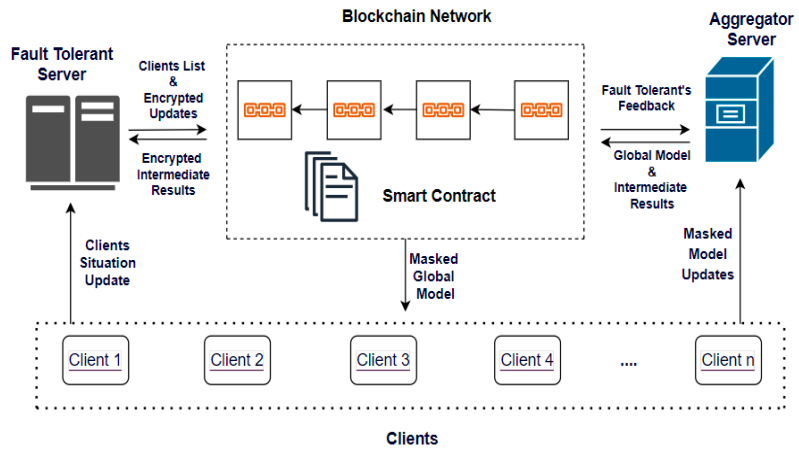

Figure 3. Architecture of the proposed framework.

5.2. Synchronization Process

The synchronization of the Blockchain with the other components in the architectural framework is in real time. Once the global model updates are aggregated by the central server, a hash of the update or summary is recorded on the Blockchain. This provides an immutable record of the model update. A consensus is reached by the validators on the validity of the recorded model updates. This record becomes a permanent part of the Blockchain. Synchronization is achieved through periodic communication between the servers and the immutable records on the Blockchain. The two servers (fault-tolerant server and aggregator server) periodically synchronize with the Blockchain to check for new records and status updates. In an unexpected failure of the central server, the Blockchain immutable records can serve as a reference point to continue the training. The Blockchain serves as a traceable, consistent, transparent, and a trusted reference point across all components in the architectural framework.

5.3. Client Situation Updates (CSUs)

In this context, stragglers and dropouts will be classified as crashed clients. To avoid infinite loops and handle crashed clients, the aggregator server does not need to wait for all the clients to return their post-trained models but will dynamically be able to implement a callback or end-process aggregation once sufficient updates have been returned. This mechanism enhances round efficiency in situations where there is a high probability of crash of clients. CSUs are carried out in a privacy-preserved manner.

Available clients = A.
Selected clients = B.
Crashed clients = C.

$$SU = \frac{|B - B \cap C|}{|A|} \tag{12}$$

After every round of training, the crash ratio is determined using the following:

$$C_r = \frac{C}{A} \tag{13}$$

where C_r is the crash ratio.

In every training round, the fault-tolerant server will send these updates to the aggregator server through the Blockchain network. The aggregator server uses these updates to dynamically evaluate when to implement a callback or end-process mechanism. These updates will enable the Blockchain network during the selection of clients for training to make the decision to allow other clients to step in and take the place of the crashed clients.

5.4. Federated Learning Loss Function

The loss function is a means of evaluating the model's performance on the data it has been trained on. The loss function used in this FL is the cross-entropy loss, a standard supervised learning loss function that examines the difference in the anticipated probability distribution and the actual probability distribution of device data. The loss function of each device is calculated, and the results are aggregated to update the global model. This ensures that the global model is a true representation of the device data. To minimize the expected loss across all devices, FedAvg is used. FedAvg is a technique where multiple devices store training data locally, and the aggregated local updates from these devices are used to train a model.

Let model parameters = w.
Number of devices = n.
Loss function = L.
Loss function for the i^{th} device = $li(w)$.
Learning rate = η.
Derivative function = ∇.

The FedAvg loss function can be expressed as follows:

$$L(w) = \frac{1}{n} * \sum_{i=1}^{n} li(w) \tag{14}$$

The FedAvg goal is to reduce the loss function based on the model parameter w. Using the following update rule, the FedAvg algorithm minimizes the loss function by iteratively updating the model parameters.

$$w_{new} = w_{old} + \eta * \nabla(L(w_{old})) \tag{15}$$

$\nabla(L(w_{old}))$ is the gradient of the loss function with respect to the model parameter w_{old}, and it is computed as the aggregated local gradients of each device.

$$\nabla(L(w_{old})) = \frac{1}{n} * \sum_{i=1}^{n} \nabla(li(w_{old})) \tag{16}$$

With respect to the model parameter w_{old}, $\nabla(li(w_{old}))$ is the gradient of the loss function for the ith device. Applying the update rule iteratively, the FedAvg algorithm minimizes the overall loss function across the devices by converging to a set of model parameters.

6. Illustration of Callback Function and End-Process Mechanism in Federated Learning

Based on the partitioning sample, the strategies that can be adopted to implement FL are Vertical Federated Leaning (VFL), Horizontal Federated Learning (HFL), and Federated Transfer Learning (FTL) [36,37].

(1) VFL: This is a scenario where the sample identities (IDs) are the same, but the sample spaces shared by the datasets are different. There is quite a huge gap in the user–space intersection due to differences in feature space. VFL is also called feature-based FL.

Let S_y , F_y, *and* L_y denote the y^{th} sample ID space, the y^{th} feature space, and the y^{th} label space, respectively. Consequently, let the data held by each data owner y be represented by the matrix My. Then, VFL can be summarized as follows:

$$S_y \ = S_z, \ F_y \neq F_z, \ L_y \neq L_z, \ \forall M_y, M_z, \ y \neq z \tag{17}$$

The z^{th} expression is the same as that of the y^{th}.

(2) HFL: In this category, data samples are different but share the same feature space. In this scenario, each device shares an identical feature space, which makes the user–space intersection less significant. HFL represents a real-life scenario, and most FL studies are based on this strategy. HFL can be summarized as follows:

$$S_y \ \neq S_z, \ F_y = F_z, \ L_y = L_z, \ \forall M_y, M_z, \ y \neq z \tag{18}$$

(3) FTL: This is a combination of the VFL and HFL strategies, and it is applicable when there are differing data samples and feature spaces of two device datasets. This can be summarized as follows:

$$S_y \ \neq S_z, \ F_y \neq F_z, \ L_y \neq L_z, \ \forall M_y, M_z, \ y \neq z \tag{19}$$

The architecture in Figure 4 utilizes a synchronous HFL strategy. The technique is to implement an end-process aggregation, called the deadline aggregation mechanism. This aggregation mechanism is dynamically implemented based on the updates from the fault-tolerant server. When the devices have returned a certain sufficient percentage of post-trained models, the server will implement a callback and issue a force stop to devices yet to return their post-trained models. On the other hand, if this sufficient percentage of the post-trained model, as specified by the FL server, is not yet achieved and a deadline has been reached, a force stop will be issued to the selected devices so that the issue of an

infinite loop or endless waiting could be avoided. The assumption made in Figure 4 is to use five devices, but in a real use case scenario, it will involve hundreds of thousands of devices. β_1, β_2, β_3, β_4, and β_5 represent the post-trained models of devices 1, 2, 3, 4, and 5, respectively, while θ_G represents the global model.

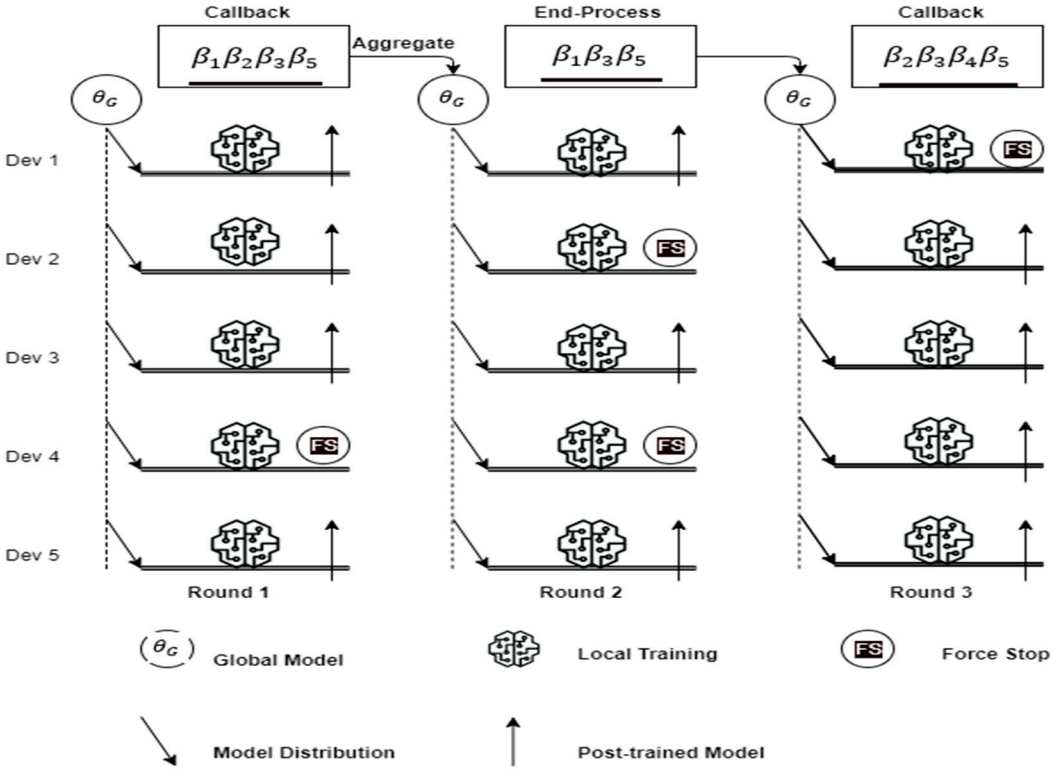

Figure 4. Example of FL model training and aggregation process.

In each round of training, the FL server, through the Blockchain network, sends a global model to all participating devices for local training. In round 1, a callback was initiated because a certain percentage of the post-trained models ($\beta_1\beta_2\beta_3\beta_5$) was returned to the server, and a force stop was issued to the remaining devices yet to return their post-trained models, in this case, device 4. As illustrated in Figure 2 and using (10), the secured aggregated masked post-trained models are unmasked by the aggregator server to produce a new global model used for the next round of training. It is worth noting that only returned post-trained models are needed for unmasking and aggregation. In the second round of training (round 2), a callback was not issued because the percentage of post-trained models, as specified by the server, was not met. At the expiration of the deadline, a force stop is issued to devices yet to return their post-trained models. The returned post-trained models ($\beta_1\beta_3\beta_5$**)** are unmasked and aggregated to produce a new global model needed for the next round of training. In the third round (round 3), a callback was issued by the server because the percentage of the returned post-trained model was met, and a force stop was issued before the deadline was reached. A new model emerges from the secure aggregation of the returned post-trained models ($\beta_2\beta_3\beta_4\beta_5$**)** of dev 2, dev 3, dev 4, and dev 5, respectively, as shown in Figure 4. The iteration continues until convergence is reached and a final global model is produced.

7. Results, Discussion, and Performance Evaluation

In this section, we conduct an experiment to demonstrate our proposed framework's performance in model update, secure aggregation, and evaluation of the trained model. This is with respect to the clients and the aggregator server.

7.1. Implementation

The model training was implemented through Python on a Linux operating system using Tensorflow Federated (TFF) libraries. FL training was carried out on a laptop configuration of 8 GB RAM, 500 GB HDD, and 1.3 GHz processor. Using TFF libraries, an FL model was trained on the EMNIST dataset. This dataset was pre-processed by defining a pre-processed function. Pre-processing steps were carried out so that the image pixels and labels could be converted into suitable format for training a keras model. For efficient training, the input data were pre-processed by reshaping the images and labels, batching the data. Using keras, a Convolutional Neural Network (CNN) model was created, which consisted of convolution, pooling, flattening, and dense layers. To enable FL, the model was wrapped with TFF.

The FedAVG process was built, specifying the model function, server, and client optimizers. The training state was initialized, and the training process was executed in multiple rounds. The model was trained repeatedly using data from various clients in a federated manner. During each round of training, the progress was monitored, and metrics such as accuracy and loss were analysed. This process continued until sufficient post-trained models (desired accuracy threshold was achieved) were returned to the server, or the maximum number of rounds was reached. After the training process, the model was evaluated on the test dataset, the test data were batched, and the accuracy was computed. Finally, the training progress was visualized by plotting the accuracy against the number of rounds and the accuracy against loss.

To achieve better convergence, we implemented a learning rate scheduler that decreased the learning rate (lr) over iterations. We achieved speedy convergence by implementing an exponential decay learning rate scheduler. Furthermore, the feedback loop of our framework enhanced speedy convergence and reduced communication cost. The exponential decay learning rate formular is as follows:

$$decayed\_lr = (init\_lr) \times (decay\_rate)^{step/decay\_steps} \tag{20}$$

where:

decayed_lr: exponential decay learning rate.
init_lr: the starting learning rate.
decay_rate: the base of the exponential function.
step: the current round of FL training.
decay_steps: the decay step.

7.2. Results

In Figure 5, we can see a graph that displays the accuracy of the model versus the number of training rounds. As the training continues, the accuracy of the model changes, providing valuable information on how well the model performs for a specific task. A closer look at the graph will identify the trend in the accuracy improvement over time. This gives an understanding of whether the model is converging or if further training is required. To hasten model convergence and avoid an infinite loop or endless waiting, a callback was initiated once enough post-trained models (desired accuracy threshold) had been returned to the server. In this case, 75% accuracy was achieved. Considering the unreliable nature of IoT (stragglers and drop-offs) and to reduce the communication cost, callback at a 75% accuracy threshold (three quarter of the clients) will guarantee a good model and could be a good feat within a resource-constrained setting.

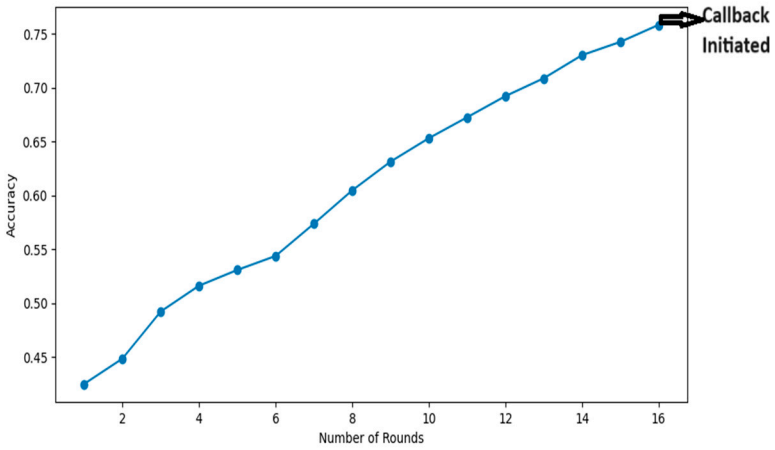

Figure 5. A plot of accuracy and number of rounds.

Figure 6 shows the relationship between the accuracy of the model and loss during the training process. The accuracy indicates how well the model is performing, while the loss indicates the ability of the model to minimize prediction loss. Using the graph, an analysis of the trade-off between accuracy and loss can be achieved. With every iteration, the model learns, and the loss should decrease, leading to improved accuracy. By a closer examination of the graph, we can identify patterns, such as decreasing loss accompanying increasing accuracy. From Figure 6, the model performance can be assessed, and it will also assist in finding an optimal balance between accuracy and loss.

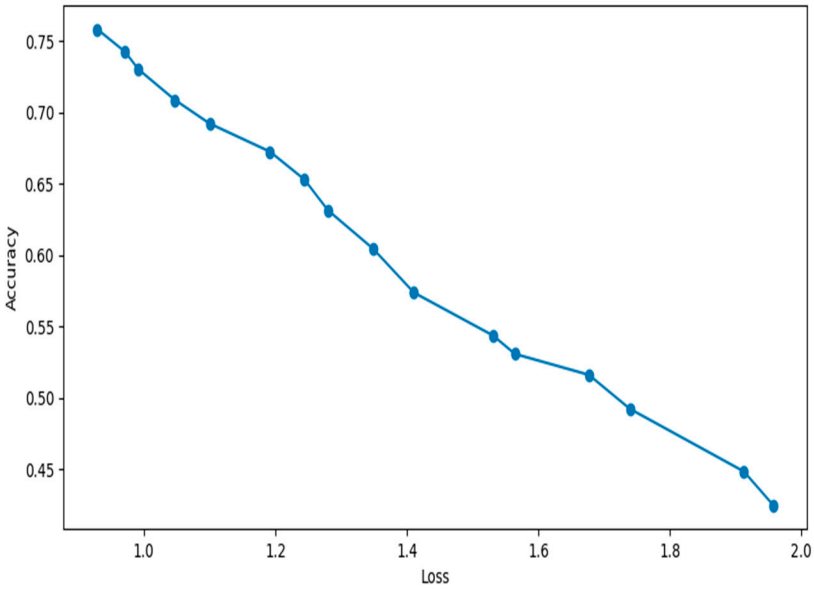

Figure 6. A plot of accuracy and loss to access the model performance at every iteration.

7.3. Comparison

Here, we compare our scheme with several existing related schemes in terms of secure aggregation and model performance. Table 1 gives a functional comparative evaluation of our scheme and previous schemes from the following viewpoints: the ability to handle dropouts, computational complexity, scalability, and communication cost. It can be observed that our scheme successfully achieved all functionalities.

Table 1. A comparative evaluation of our scheme and previous schemes.

Schemes	Handle Dropouts	Computational Complexity	Scalable	Communication Cost
[7]	YES	NO	NO	Low
[15]	YES	NO	NO	Low
[16]	NO	YES	NO	Low
[17]	NO	NO	NO	High
[21]	NO	YES	NO	High
[24]	NO	YES	NO	High
[25]	NO	YES	NO	High
OURS	YES	NO	YES	Low

In handling stragglers and dropouts, the client situation updates are communicated to the fault-tolerant server in a privacy-preserved manner for each round of training. These updates are used within the network to estimate the training time and make decisions on when to end the training process of each round to avoid an infinite loop. These updates will assist in deciding on the number of clients to select for each round of training. If there are more dropouts in each round, the tendency to achieve an accuracy threshold (initiating callback) will be low. The aggregator server uses these updates to set an end-process time if a callback is not yet initiated.

As shown in Figure 7, the computational cost is a measure of the time it takes for each training round. It is a measure of the difference between the start and end times of the training process for each round. The computational complexity of our scheme is not high because the aggregation of the returned trained models is carried out using the aggregator server rather than the Blockchain network. The computational complexity associated with aggregating each round of the training using the Blockchain is eliminated. Here, we retain the central server's ability to aggregate returned trained models to eliminate the complexity associated with aggregation using the Blockchain network. Figure 7 shows the performance of our scheme in terms of communication round complexity in seconds, and it needs just two rounds of communication. The first one is the clients sending their model updates to the aggregator server for aggregation, while the second is sending their status update to the fault-tolerant server for managing dropouts and stragglers. Our communication round can only be compared to Ref. [7], which made use of two communication rounds.

Figures 8 and 9 illustrate the scalability tests of accuracy and loss, respectively. They show various curves that represent the accuracy and loss of models trained with varying number of clients, such as 50, 100, 150, 200, and 250 clients. Across training rounds, the accuracy and loss metrics change when there is variation in the number of clients available for training. After several training rounds and across all client counts, the accuracy generally increases and stabilizes, while the loss decreases. The early training rounds of both figures indicate a period of rapid correction or adjustment according to Equation (20) in the FL process before stabilization. This correction or adjustment is also based on the feedback loop of our architectural framework. From a careful observation of our architectural framework and previous schemes in Table 1, it appears that only our scheme has a feedback loop after every communication round. The feedback loop of our framework provides the status update and performance of the clients in a privacy-preserved manner. This gives an overview of the training process and ensures that more clients are introduced in the next round of training when there are more stragglers and dropouts in the preceding round. Across different numbers of clients in both figures, the variability in curves does not show drastic differences. Rather, it indicates that the model is relatively stable despite an increase in the number of clients. An increase in the number of clients did not result in a dramatic decrease in accuracy or a considerable increase in loss. Consequently, when the number of clients is increased, the accuracy and loss seem to stabilize. The accuracy curves indicate

that the training stability converged as the number of clients increases; it remained stable until the desired accuracy was attained or callback was implemented. Our framework shows positive scalability traits across varying client distributions. This indicates that our framework is capable of handling more clients without degradation in performance.

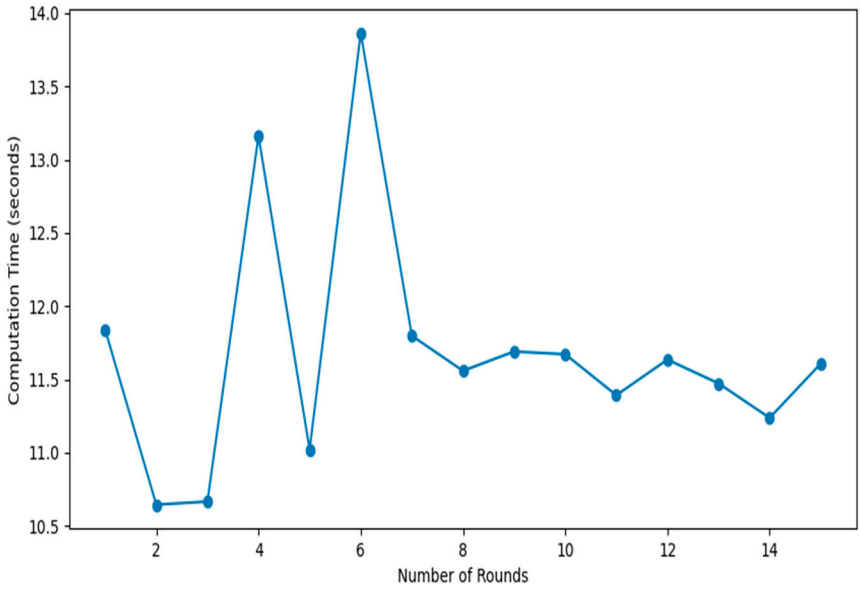

Figure 7. Computational cost of model updates performed in each round.

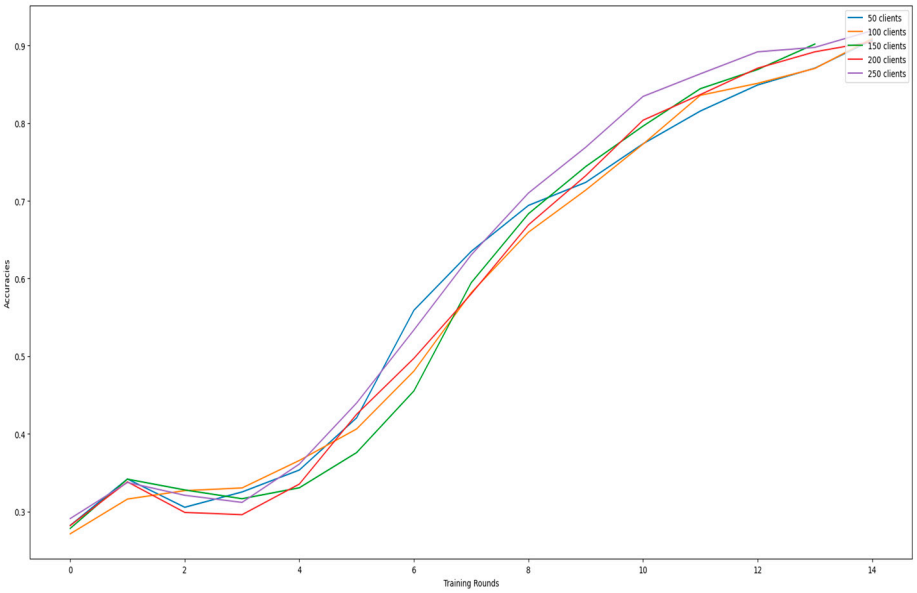

Figure 8. Scalability test of accuracy.

In a scalability assessment that considers factors such as fault tolerance and secure aggregation strategies, our scheme's architectural framework (Figure 3) indicates that secure aggregation could be carried out using the aggregator (central) server or fault-tolerant server. It could also be reconfigured to use the Blockchain network for aggregation. This distinguishes our scheme from previous schemes in terms of fault tolerance and secure aggregation strategies. The constant communication between the aggregator server and Blockchain network ensures that the latest updates are stored in the Blockchain network. Therefore, the abrupt failure of the aggregator server will not lead to a total collapse of the training process because the Blockchain network or fault-tolerant server could continue

orchestrating the training process from the point of a sudden or unforeseen breakdown of the aggregator server. Our framework prevents the loss of training time and wastage of resources due to the unexpected failure of the central server.

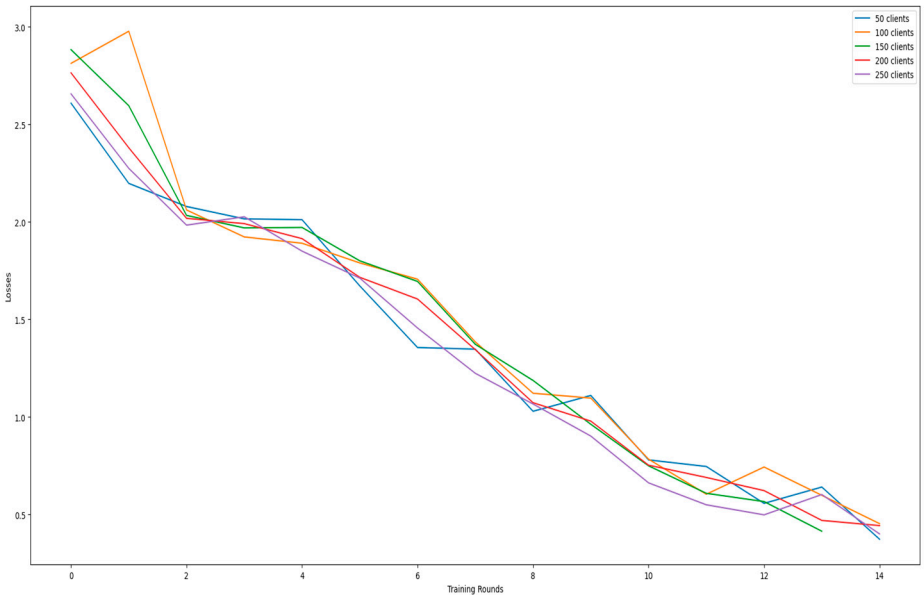

Figure 9. Scalability test of loss.

Finally, our scheme's trust, transparency, and privacy-preservation mechanism enable adequate participation in the model training. With more active clients participating in the model training, speedy convergence of the global model is achieved, the number of iterations (training time) is reduced, and communication costs are low because of callback or the end-process mechanism of aggregation.

8. Conclusions

In this article, a secured Blockchain-based aggregation mechanism using FL has been proposed. The model utilizes Blockchain technology as a means of masking local data of end devices within an FL framework to avoid the reconstruction of local data by a malicious server. Blockchain as a distributed ledger offers an immutable data structure, enhanced security, faster settlement in payment systems, and a consensus protocol. Blockchain provides a unique system of accumulating data in a chronological and privacy-preserved manner because the greatest resource of any organization is data. Using Blockchain, the model tends to solve the threat of SPOF, ensures the privacy preservation of end devices' local data, eliminates the issues of vulnerability, and ensures trust, transparency, fairness, and security. The callback or end-process mechanism eliminates the need for endless waiting that could arise because of the issue of stragglers and dropouts, which is evident in the FL scenario. The secure aggregation mechanism of the masked models using a callback function/end-process technique ensures that only returned post-trained models are needed for aggregation. This mechanism prevents the problem of complicated handling of local masked models of stragglers and dropouts by the FL server during aggregation to produce a global model. The framework enables immutable, efficient, transparent, and secure communication between the FL server and devices within the federated network. The limitation of Blockchain technology, as observed in most research, is the computational complexity in maintaining security and privacy preservation when there is increase in the network. With this technique, the computational complexity is greatly reduced because the Blockchain network does not perform aggregation; rather, the aggregator server aggregates the returned post-trained model. Finally, only returned post-trained models are needed for aggregation, which benefits resource-constrained IoT devices.

Author Contributions: Writing—original draft, W.E.M.; Writing—review & editing, C.M. and G.E.; Supervision, C.M. and G.E. All authors have read and agreed to the published version of the manuscript.

Funding: EP/R007195/1 (Academic Centre of Excellence in Cyber Security Research—University of Warwick); EP/N510129/1 (The Alan Turing Institute); and EP/S035362/1 (PETRAS National Centre of Excellence for IoT Systems Cybersecurity).

Data Availability Statement: Data available on request.

Acknowledgments: This work has been supported by UKRI through the grants: Academic Centre of Excellence in Cyber Security Research—University of Warwick (EP/R007195/1), The Alan Turing Institute (EP/N510129/1), and PETRAS National Centre of Excellence for IoT Systems Cybersecurity (EP/S035362/1). We also acknowledge the constructive criticisms, feedback, and suggestions from the reviewers, which have improved the manuscript. We are also grateful to colleagues whose expertise, insights, and suggestions shaped this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hussain, G.K.J.; Manoj, G. Federated Learning: A Survey of a New Approach to Machine Learning. In Proceedings of the 2022 1st International Conference on Electrical, Electronics, Information and Communication Technologies, ICEEICT 2022, Trichy, India, 16–18 February 2022; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2022. [CrossRef]
2. Abdulrahman, S.; Tout, H.; Ould-Slimane, H.; Mourad, A.; Talhi, C.; Guizani, M. A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. *IEEE Internet Things J.* **2021**, *8*, 5476–5497. [CrossRef]
3. Wang, S.; Sahay, R.; Brinton, C.G. How Potent Are Evasion Attacks for Poisoning Federated Learning-Based Signal Classifiers? 2023. Available online: http://arxiv.org/abs/2301.08866 (accessed on 22 July 2023).
4. Rahman, K.M.J.; Ahmed, F.; Akhter, N.; Hasan, M.; Amin, R.; Aziz, K.E.; Islam, A.K.M.M.; Mukta, S.H. Challenges, Applications and Design Aspects of Federated Learning: A Survey. *IEEE Access* **2021**, *9*, 124682–124700. [CrossRef]
5. Chen, H.; Asif, S.A.; Park, J.; Shen, C.-C.; Bennis, M. Robust Blockchained Federated Learning with Model Validation and Proof-of-Stake Inspired Consensus. 2021. Available online: www.aaai.org (accessed on 22 July 2023).
6. Bhatia, L.; Samet, S. Decentralized Federated Learning: A Comprehensive Survey and a New Blockchain-based Data Evaluation Scheme. In Proceedings of the 2022 4th International Conference on Blockchain Computing and Applications, BCCA 2022, San Antonio, TX, USA, 5–7 September 2022; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2022; pp. 289–296. [CrossRef]
7. Huang, C.; Yao, Y.; Zhang, X.; Teng, D.; Wang, Y.; Zhou, L. Robust Secure Aggregation with Lightweight Verification for Federated Learning. In Proceedings of the 2022 IEEE 21st International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2022, Wuhan, China, 9–11 December 2022; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2022; pp. 582–589. [CrossRef]
8. Liu, P.; Xu, X.; Wang, W. Threats, attacks and defenses to federated learning: Issues, taxonomy and perspectives. *Cybersecurity* **2022**, *5*, 4. [CrossRef]
9. Li, D.; Han, D.; Weng, T.-H.; Zheng, Z.; Li, H.; Liu, H.; Castiglione, A.; Li, K.-C. Blockchain for federated learning toward secure distributed machine learning systems: A systemic survey. *Soft Comput.* **2022**, *26*, 4423–4440. [CrossRef] [PubMed]
10. Salim, S.; Turnbull, B.; Moustafa, N. A Blockchain-Enabled Explainable Federated Learning for Securing Internet-of-Things-Based Social Media 3.0 Networks. *IEEE Trans. Comput. Soc. Syst.* **2021**, 1–17. [CrossRef]
11. Manvith, V.S.; Saraswathi, R.V.; Vasavi, R. A performance comparison of machine learning approaches on intrusion detection dataset. In Proceedings of the 3rd International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, ICICV 2021, Tirunelveli, India, 4–6 February 2021; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2021; pp. 782–788. [CrossRef]
12. Qu, Y.; Pokhrel, S.R.; Garg, S.; Gao, L.; Xiang, Y. A Blockchained Federated Learning Framework for Cognitive Computing in Industry 4.0 Networks. *IEEE Trans. Ind. Inform.* **2021**, *17*, 2964–2973. [CrossRef]
13. Passerat-Palmbach, J.; Farnan, T.; McCoy, M.; Harris, J.D.; Manion, S.T.; Flannery, H.L.; Gleim, B. Blockchain-orchestrated machine learning for privacy preserving federated learning in electronic health data. In Proceedings of the 2020 IEEE International Conference on Blockchain, Blockchain 2020, Rhodes, Greece, 2–6 November 2020; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2020; pp. 550–555. [CrossRef]
14. Ullah, I.; Deng, X.; Pei, X.; Jiang, P.; Mushtaq, H. A verifiable and privacy-preserving blockchain-based federated learning approach. *Peer Peer Netw. Appl.* **2023**, *16*, 2256–2270. [CrossRef]
15. Fereidooni, H.; Marchal, S.; Miettinen, M.; Mirhoseini, A.; Mollering, H.; Nguyen, T.D.; Rieger, P.; Sadeghi, A.-R.; Schneider, T.; Yalame, H.; et al. SAFELearn: Secure Aggregation for private FEderated Learning. In Proceedings of the 2021 IEEE Symposium on Security and Privacy Workshops, SPW 2021, San Francisco, CA, USA, 27 May 2021; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2021; pp. 56–62. [CrossRef]

16. Wu, D.; Pan, M.; Xu, Z.; Zhang, Y.; Han, Z. Towards Efficient Secure Aggregation for Model Update in Federated Learning. In Proceedings of the 2020 IEEE Global Communications Conference, GLOBECOM 2020—Proceedings, Taipei, Taiwan, 7–11 December 2020; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2020. [CrossRef]
17. Zhao, L.; Jiang, J.; Feng, B.; Wang, Q.; Shen, C.; Li, Q. SEAR: Secure and Efficient Aggregation for Byzantine-Robust Federated Learning. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 3329–3342. [CrossRef]
18. Pokhrel, S.R.; Choi, J. Federated Learning with Blockchain for Autonomous Vehicles: Analysis and Design Challenges. *IEEE Trans. Commun.* **2020**, *68*, 4734–4746. [CrossRef]
19. Guo, X. Implementation of a Blockchain-enabled Federated Learning Model that Supports Security and Privacy Comparisons. In Proceedings of the 2022 IEEE 5th International Conference on Information Systems and Computer Aided Education, ICISCAE 2022, Dalian, China, 23–25 September 2022; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2022; pp. 243–247. [CrossRef]
20. Zhang, P.; Liu, G.; Chen, Z.; Guo, J.; Liu, P. A study of a federated learning framework based on the interstellar file system and blockchain: Private Blockchain Federated Learning. In Proceedings of the 2022 3rd International Conference on Computer Vision, Image and Deep Learning and International Conference on Computer Engineering and Applications, CVIDL and ICCEA 2022, Changchun, China, 20–22 May 2022; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2022; pp. 267–273. [CrossRef]
21. Zhang, Q.; Palacharla, P.; Sekiya, M.; Suga, J.; Katagiri, T. Blockchain-based Secure Aggregation for Federated Learning with a Traffic Prediction Use Case. In Proceedings of the 2021 IEEE Conference on Network Softwarization: Accelerating Network Softwarization in the Cognitive Age, NetSoft 2021, Tokyo, Japan, 28 June–2 July 2021; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2021; pp. 372–374. [CrossRef]
22. Sun, Y.; Esaki, H.; Ochiai, H. Blockchain-Based Federated Learning against End-Point Adversarial Data Corruption. In Proceedings of the 19th IEEE International Conference on Machine Learning and Applications, ICMLA 2020, Miami, FL, USA, 14–17 December 2020; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2020; pp. 729–734. [CrossRef]
23. Al Mallah, R.; López, D.; Halabi, T. Blockchain-enabled Efficient and Secure Federated Learning in IoT and Edge Computing Networks. In Proceedings of the 2023 International Conference on Computing, Networking and Communications, ICNC 2023, Honolulu, HI, USA, 20–22 February 2023; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2023; pp. 511–515. [CrossRef]
24. Kalapaaking, A.P.; Khalil, I.; Rahman, M.S.; Atiquzzaman, M.; Yi, X.; Almashor, M. Blockchain-based Federated Learning with Secure Aggregation in Trusted Execution Environment for Internet-of-Things. *IEEE Trans. Ind. Inform.* **2023**, *19*, 1703–1714. [CrossRef]
25. Chen, Y.; Lin, F.; Chen, Z.; Tang, C.; Jia, R.; Li, M. Blockchain-based Federated Learning with Contribution-Weighted Aggregation for Medical Data Modeling. In Proceedings of the 2022 IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems, MASS 2022, Denver, CO, USA, 19–23 October 2022; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2022; pp. 606–612. [CrossRef]
26. Li, Y.; Chen, C.; Liu, N.; Huang, H.; Zheng, Z.; Yan, Q. A Blockchain-Based Decentralized Federated Learning Framework with Committee Consensus. *IEEE Netw.* **2021**, *35*, 234–241. [CrossRef]
27. Miao, Y.; Liu, Z.; Li, H.; Choo, K.K.R.; Deng, R.H. Privacy-Preserving Byzantine-Robust Federated Learning via Blockchain Systems. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 2848–2861. [CrossRef]
28. Adhikari, N.; Ramkumar, M. IoT and Blockchain Integration: Applications, Opportunities, and Challenges. *Network* **2023**, *3*, 115–141. [CrossRef]
29. Zhang, C.; Xu, Y.; Hu, Y.; Wu, J.; Ren, J.; Zhang, Y. A Blockchain-Based Multi-Cloud Storage Data Auditing Scheme to Locate Faults. *IEEE Trans. Cloud Comput.* **2022**, *10*, 2252–2263. [CrossRef]
30. Li, C.; Yuan, Y.; Wang, F.Y. Blockchain-enabled federated learning: A survey. In Proceedings of the 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence, DTPI 2021, Beijing, China, 15 July–15 August 2021; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2021; pp. 286–289. [CrossRef]
31. Christina, K.; Kesavamoorthy, R. Evolution of Blockchain and Smart Contracts: A State of the Art Review. In Proceedings of the 2023 International Conference on Intelligent Systems for Communication, IoT and Security, ICISCoIS 2023, Coimbatore, India, 9–11 February 2023; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2023; pp. 235–240. [CrossRef]
32. Xu, Y.; Zhang, C.; Zeng, Q.; Wang, G.; Ren, J.; Zhang, Y. Blockchain-Enabled Accountability Mechanism against Information Leakage in Vertical Industry Services. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 1202–1213. [CrossRef]
33. Sadath, L.; Mehrotra, D.; Kumar, A. Scalability in Blockchain—Hyperledger Fabric and Hierarchical Model. In Proceedings of the 2022 IEEE Global Conference on Computing, Power and Communication Technologies, GlobConPT 2022, New Delhi, India, 23–25 September 2022; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2022. [CrossRef]
34. Le, H.T.; Nguyen, T.T.L.; Nguyen, T.A.; Ha, X.S.; Duong-Trung, N. BloodChain: A Blood Donation Network Managed by Blockchain Technologies. *Network* **2022**, *2*, 21–35. [CrossRef]
35. Ahmed, M.R.; Meenakshi, K.; Obaidat, M.S.; Amin, R.; Vijayakumar, P. Blockchain Based Architecture and Solution for Secure Digital Payment System. In Proceedings of the IEEE International Conference on Communications, Montreal, QC, Canada, 14–23 June 2021; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2021. [CrossRef]

36. Imteaj, A.; Thakker, U.; Wang, S.; Li, J.; Amini, M.H. A Survey on Federated Learning for Resource-Constrained IoT Devices. *IEEE Internet Things J.* **2022**, *9*, 1–24. [CrossRef]
37. Jia, B.; Zhang, X.; Liu, J.; Zhang, Y.; Huang, K.; Liang, Y. Blockchain-Enabled Federated Learning Data Protection Aggregation Scheme With Differential Privacy and Homomorphic Encryption in IIoT. *IEEE Trans. Ind. Inform.* **2022**, *18*, 4049–4058. [CrossRef]