IoT Data Security: An Integration of Blockchain and Federated Learning

用IPFS来存储local model，然后将IPFS的存储hash上传到blockchain上。why？技术细节没有，炒作概念

Gagandeep Shubham*, Vidushi Agarwal†, and Sujata Pal‡
Department of Computer Science and Engineering
Indian Institute of Technology Ropar, Ropar, India
* 2020csm1009@iitrpr.ac.in
† vidushi.19csz0010@iitrpr.ac.in
‡ sujata@iitrpr.ac.in

***Abstract*—Technological advancement has led to a rapid increase in the growth of IoT devices leading to a vast amount of generated data. Manufacturers of such devices utilize machine learning algorithms to extract valuable insights from user data. However, this can give rise to critical issues surrounding data leakage and privacy. To tackle these issues, utilizing blockchain as a decentralized database to securely store data and employing federated learning to extract useful insights from user data can provide a viable solution. In this paper, we propose a three-layered, decentralized architecture that uses a traditional federated learning mechanism in conjunction with the Ethereum blockchain. Moreover, for data management, we use InterPlanetary File System (IPFS) which is a peer-to-peer network used to store data in a decentralized manner. We tested our model's feasibility by using CIFAR-10 dataset and Python as the programming language with a framework for federated learning on a general purpose computer. We used Ganache_v2.5.4 and Truffle_v5.4.22 for developing smart contracts and testing and deploying them over the Ethereum blockchain.**

***Index Terms*—Blockchain, IPFS, federated learning, smart contracts, security.**

I. Introduction

With the frequent technological advancements in today's day and age, data has become such an essential resource that it needs to be protected at all costs. There has been an exponential increase in the growth of IoT based widgets such as wearable devices, smart home devices, smartwatches, refrigerators, and air conditioners that are capable of storing data with a minimal capacity on their built-in storage or over the cloud [1]. Considering the possibility of malicious entities exploiting collected data to gain access to users's personal information, it is crucial to ensure the data is adequately secured. Manufacturers or industries typically aggregate all raw data at a central server and utilize it to train machine learning models. However, this central server can be susceptible to security threats that could potentially compromise users' privacy. Since all the users don't want to share their personal data, the manufacturers can adopt federated learning techniques instead of traditional machine learning techniques to maintain user privacy. Raw data remains on the user's device in federated learning [2], and only the parameter values acquired through local model training are transmitted to the global server. These values are subsequently aggregated to create a global model. This deep learning technique eliminates the need to share raw data with the server. However, the concept of central authority would still persist. Therefore, we introduce the concept of blockchain to reduce the influence of a central authority. Blockchain could be one solution to the aforementioned data security problems. Being a decentralized technology, it can provide a way to store the data as well as protect it from security threats [3]. Decentralizing data not only helps in resolving security issues but it also makes the network fault tolerant as multiple nodes are involved in storing the information. By utilizing blockchain and federated learning, the need for a central authority can be eliminated, thereby mitigating the risks of data leakage and other security breaches. Within the decentralized network of blockchain and FL, each participant possesses a copy of the ledger, as demonstrated in Figure 1.

Therefore, the major contributions of this work as highlighted as follows:

1) We leverage federated learning and blockchain to train our model along with Interplanetary File System (IPFS) to provide decentralized storage for model updates.
2) We incorporate smart contracts to automate the process of storing and retrieving model updates to and from Ethereum blockchain respectively.
3) The evaluation results show that the machine learning model trained using federated learning while incorporating the decentralized storage, encryption and blockchain observe an accuracy of 77.49%, which is comparable to the traditionally trained machine learning model having an accuracy 78.38% with no security.

Section II presents a brief description of the related and existing works. In section III, we present our approach and the three-layered security architecture. Section IV discusses the implementation detail more elaborately with some experimental results. Finally, we address the conclusion and future works in section V.

II. Related Work

This sections briefs the state-of-the-art work done by various researchers in the fields of blockchain and federated learning. Ghimire at al. [4] proposed a survey that discusses the major ongoing research challenges, trends and efforts in the field of federated learning for IoT. Dinh et al. [5] proposed a protection method that uses differential privacy to protect location data privacy, without reducing the utility of data in

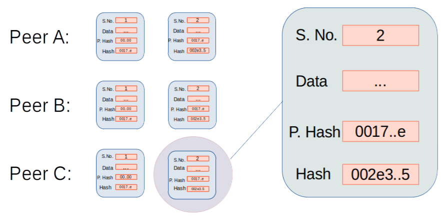


Fig. 1: A distributed network of blockchain.

Industrial IoT (IIoT). Zyskind et al. [6] created a protocol to use blockchain as an automated access-control manager, allowing users to retain control and ownership of their data. Liu et al. [7] proposed a deep reinforcement learning-based blockchain-enabled efficient data collection and secure sharing method to establish a safe and dependable environment.

Amongst these works, consensus protocol is a core component that affects the performance of the model to a great extent. The commonly used protocol is Proof of Work (PoW) that requires extensive resources for solving the mathematical puzzles thus limiting the application area of the above frameworks. Agarwal et al. [8] proposed a scalable architecture of blockchain for IoT networks that uses different consensus mechanisms and off-chain solutions for storing the data on blockchain. Bacon et al. [9] presented a federated learning strategy allowing many data owners to collaborate on training a global model without releasing their raw data in order to protect user privacy. Geyer et al. [10] proposed an algorithm for client-sided differential privacy-preserving federated optimization to hide client contributions throughout the training phase. Yu et al. [11] presented a proactive content caching strategy based on federated learning, based on a hierarchical design in which the server aggregates the users' training updates. To remedy the issue of the centralization in most of the above mentioned works, Lu et al. [12] proposed a framework that shares the cached machine learning model with the data requesters. However, the lack of encryption techniques makes it vulnerable to cyber attacks such as man-in-the-middle attack. Arachchige et al. [13] proposed a framework called PriModChain that blends various modules such as FedML, Ethereum Blockchain, and smart contracts to provide security to IIoT data. Passerat-Palmbach et al. [14] combined blockchain and federated learning with encryption techniques in order to provide security to electronic health data. Zhang et al. [15] introduced edge computing into IIot to provide high computation power so that the devices can perform the tasks of training and updating the models in federated learning. Su et al. [16] proposed a model that combines blockchain technology and a proxy re-encryption technology to provide a secure data sharing process. TrustChain was proposed in [17] that incorporates consortium blockchain to track and trace the complete supply chain and the interaction amongst its participants.

In most of the existing data sharing schemes, the presence of a centralized curator increases the risk of data leakage. Therefore, we implement our model using the distributed Ethereum blockchain technology. The evaluation results show that the cost and the latency incurred during the whole process is minimal as compared to the traditionally trained ML models with no security.

III. Proposed Architecture

A. System Model

To protect the privacy of users' data, we utilize the concept of blockchain combined with federated learning rather than uploading raw data to the blockchain. Our end goal is to provide the manufacturers the information they require, i.e., useful insights from users' data to provide better services in the future while ensuring that personal data remains protected too.

In the proposed framework, we use federated learning (FL) such that initially each device trains the local model with its local data and then the learned parameters are stored in a file. Next, we leverage the distributed network provided by IPFS (InterPlanetary File System) for storing the files and every file is tagged with a unique content identifier. IPFS is a distributed peer-to-peer network created by Juan Benet [18], that provides a way to store and share data in a completely decentralized manner. All the unique content identifiers, also known as hash values, are then uploaded to the blockchain.

The proposed architecture comprises three layers, with each layer consisting of one or more entities. The architecture can be depicted through Figure 2 and is described as follows:

- Layer 1, being the bottom-most layer, consists of N IoT devices. We consider that each of these devices possess the ability to connect to the internet and is also capable of training an ML model.
- Layer 2 incorporates a peer-to-peer protocol service known as InterPlanetary File System (IPFS).
- The topmost layer, i.e., layer 3, is a blockchain network.

1) Layer 1 (IoT Layer): The IoT layer consists of N independent devices $(D_1, \ldots, D_n)$ having a capability to connect with the internet. These devices either come with a small storage capacity, or they store data over the cloud.

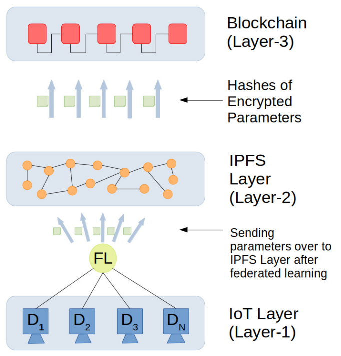
2) Layer 2 (IPFS): IPFS is a protocol that allows us to store and share data in a distributed peer-to-peer network. The advantage of this protocol over other traditional client-server architectures is that it provides fault tolerance as well as decentralization.

3) Layer 3 (Blockchain Layer): The Blockchain layer provides a secure way to store the data. Usually, storing huge amounts of data over a blockchain would require very high amount of resources, therefore, we store only the hashes of the required data in the blockchain.

All the layers mentioned above work interdependently.

B. Proposed Algorithm

Unlike traditional ML process where the data needs to be gathered at a central location, we use the FL technique



三层结构

Fig. 2: Proposed 3 Layered Architecture

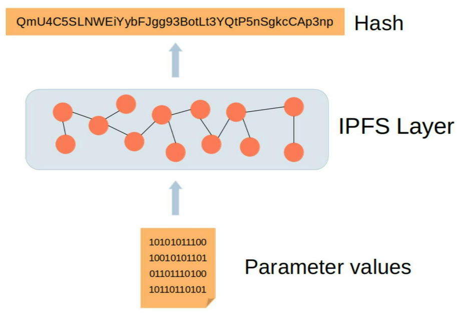


Fig. 3: Data Conversion Process by IPFS

to train the models. In order to implement the FL training, we incorporate an open source framework known as flower developed by Beutel et al. [19]. This framework helps in the training of ML models in a federated setting such that the clients' data stay on their device itself. The FL training process proceeds through a number of rounds, also known as communication rounds. Initially, a server creates a global model and distributes it among N clients participating in the FL process. Each N device begins training the initial model with their data and returns the parameter values to the server after the initial round of training. After the first round, the server aggregates all the model updates and averages them to update the global model. The exact process repeats for a number of pre-defined communication rounds. With each communication round, we send the learned parameters to the server as well as save them in encrypted format. The encrypted model updates are saved using IPFS that stores the hash as shown in Figure 3.

IPFS provides a unique content identifier i.e., hash of the data stored in a file. It distributes the files stored over it to the nodes participating in the network. Whenever someone needs to access the file, it requires the same content identifier to identify the file based on the content. However, this presents a serious issue because anyone having the content identifier could access the files and potentially harm our learning process and the personal data of the users as well.

Therefore, to prevent unauthorized access of files, we store all the content identifiers over the Ethereum blockchain using

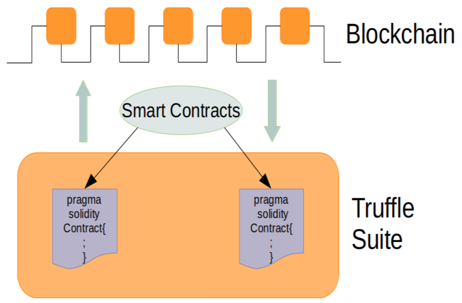


Fig. 4: Interaction with blockchain using smart contracts

blockchain开发套件

Ganache, which is a trusted high-end development environment for developing decentralized application and smart contracts. The smart contract is a self-executable piece of code, that executes whenever a new content identifier becomes available. For communicating with the blockchain, we use a smart contract as shown in Figure 4 to store all the content identifiers. The Ethereum blockchain uses Truffle and one other component of the Truffle Suite environment called Ganache for execution.

Each smart contract execution leads to a transaction over the blockchain. This transaction will have a corresponding transaction hash that can be used to verify it over the blockchain. A user/device first has to get authenticated with the blockchain to access these transaction hashes. We also use another smart contract to access the stored data hashes from the blockchain that can be retrieved by using the corresponding transaction hash. The whole mechanism is represented as a compact algorithm as depicted in Algorithm 1.

IV. Experimental Results and Analysis

This section presents the results of our proposed architecture and a comparative analysis with the traditional ML process. We use the popular image dataset called CIFAR-10 [20] for our simulation environment which contains 60,000 images spread across 10 different classes, with each class containing 6000 images. It contains 50,000 images in the training dataset, while the testing dataset contains 10,000 images.

A. Experimental Setup

We begin the process by training two ML models. They are created through PyTorch library, a simple Convolutional Neural Network (CNN) classifier with two convolution layers, one pooling layer, and three fully connected layers. The architecture of CNN is same for both the training processes. Since every image in the dataset has 3 channels, namely red, green, and blue, the image size is 32$\times$32. The input to the first convolutional layer has 3 channels and 6 filters resulting from convolution with size 3$\times$5$\times$5 where we use the stride value as 1. To reduce the number of features from feature map, we have used a kernel of size 2$\times$2 and stride set to 2 using max-pooling while subsampling. The second convolutional layer takes 6 channels as input and convolves 16 filters with 6$\times$5$\times$5 as the size of each filter.

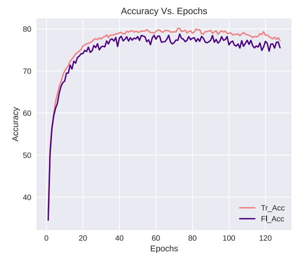
Algorithm 1 Data Storage process

```
1: for $i \leftarrow 1, N$ do
2:     Authenticate device $D_i$ with blockchain
3:     Store $< p_{b_i}, p_{r_i} >$
4:     Store $p_{b_i}$ in $ACL_i$
5: end for
6: for $i \leftarrow 1, k$(Communication rounds) do
7:     for $i \leftarrow 1, N$ do
8:         $D_i$ downloads $G_{in}$ from the server and updates $LM_{D_i}$
9:         Train $LM_{D_i}$
10:        $LM^e_{D_i} = E_{p_{r_i}} < LM_{D_i} >$
11:        Transfer $< LM^e_{D_i}, p_{b_i} >$ to the server.
12:    end for
13:    for $i \leftarrow 1, N$ do
14:        $LM^d_{D_i} = D_{p_{b_i}} < LM_{D_i} >$
15:    end for
16:    Average all parameters $\{LM^d_{D_1}, \ldots, LM^d_{D_N}\}$ as $G_{avg}$
17:    Update $G_{in}$ and transfer to devices.
18:    for $i \leftarrow 1, N$ do
19:        Transfer $LM^d_{D_i}$ to IPFS
20:        Store $H_i = Hash(LM^d_{D_i})$
21:    end for
22:    Transfer and Store $\{H_1, \ldots, H_N\}$ to blockchain
23: end for
```

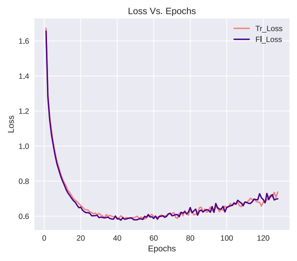
While doing subsampling for the second time, we have kept the kernel size and the stride to be same as before. Rectified Linear Unit (ReLU) activation function with 120 nodes has been used at the first fully-connected layer. The second fully-connected layer also uses the ReLU activation function with 84 nodes in total. The output layer of the network has 10 nodes and uses softmax as the activation function. Cross-entropy is used as a loss function with stochastic gradient descent (SGD) as the optimizer.

We train both the models for 128 epochs and batch size 4 for the training processes. For preparing the FL model, we leverage the flower framework's convenience class called NumPyClient. We trained the FL model for 5 communication rounds and two clients. Each client device is simulated in a different terminal window of the workstation, while the server is simulated in another terminal window of the same system. To update the global model, the server aggregates and applies an averaging strategy on the returned parameters. However, before this step, the parameter values get digitally signed using the client's private key received during the initial authentication with blockchain. This step is done to prevent against attacks such as source repudiation [21].

The server can easily decrypt the received files using the device's public keys. Usually, the server saves the parameters when a particular training round concludes, but it could lead to centralization, i.e., a single point of failure. To resolve this, instead of saving the parameters on server, we store them over IPFS using python's IPFS_API library. The IPFS returns a unique content identifier, precisely, a hash through which those files can be accessed. Next, we set up a personal blockchain using ganache. Since the hashes are sensitive, we deploy a smart contract written in the solidity programming language using truffle on the Ethereum blockchain and then use the Python's Web3 library to export the hashes over the blockchain.



(a) Accuracy comparison for traditional machine learning and federated learning



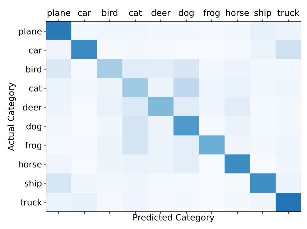
(b) Loss comparison for traditional machine learning and federated learning

Fig. 5: Performance comparison for traditional machine learning and federated learning.

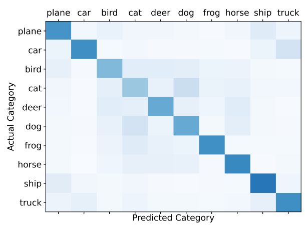
B. Simulation Results

The graphs depicted by Figures 5a and 5b show the accuracy and loss comparison for both the learning processes with 128 epochs. The tradtional training of the CNN model revealed slightly higher accuracy as compared to our FL model, however, it can be seen in Figure 5a that both the models converge at 78% accuracy. Figure 5b depicts the comparison between losses for both the training processes and it shows that they converge after 128 epochs. One of the major reasons behind this observation is the fact that the underlying architectures of the model for both traditional machine learning process and FL process are the same.

Figure 6 depicts that the classes are more accurately predicted using traditionally trained CNN model as compared to the federatedly trained CNN model. Figure 6a shows that the class "car" has been predicted more accurately as compared to the other classes, whereas from Figure 6b, we observe a light blue color for the same "car" class. This shows that despite of training the models on different clients, the accuracy of both the models does not show any significant differences. Some of the training results of our model as compared to the traditionally trained CNN model for different number of epochs are shown in Table I. The notations used are defined in Table II. From Table I, it can be observed that the accuracy for traditional training when trained with 32 epochs is 58.20% whereas the training accuracy for the FL process is 56.37%. Moreover, the execution time for the traditional training comes out to be 112s which is good in comparison with the model trained with FL. When we train the model with 64 epochs while keeping the number of training rounds same for the process, the training accuracy shows a slight increase for traditional training, i.e., 69.13% and the accuracy for the federated learning process comes out to be 68.22%. Also, the



(a) Confusion matrix for traditionally trained CNN model



(b) Confusion matrix for federatedly trained CNN model

Fig. 6: Comparison of Confusion matrix.

execution time incurs a slight increase which is recorded as 197s for traditional training and 209s for the federated training. For 128 epochs, the accuracy observed for the traditional training and federated training comes out to be almost similar, which is 78.38% and 77.49% respectively. The execution times for both the processes also incur a significant increase owing to the increased number of epochs, which is 418s and 436s for the traditional training and federated training respectively.

TABLE I: Experimental results for different number of Epochs.

Epochs	Acc_{tr}	Acc_{fl}	$Exec_{tr}$	$Exec_{fl}$
32	58.20%	56.37%	112s	134s
64	69.13%	68.22%	197s	209s
128	78.38%	77.49%	418s	436s

C. Accuracy against different number of clients and communication rounds for federated learning

We then analyze the results by training the ML models while keeping the number of epochs fixed to 32. As shown in Figure 7, the accuracy of the model observes an increment as the number of rounds increase. The accuracy of the model varies with the number of communication federation rounds, data, and the choice of ML model.

TABLE II: Notations

Notations used in Table 1	
Notation	Definition
Acc_{tr}	Traditional model accuracy
Acc_{fl}	Federated model accuracy
$Exec_{tr}$	Execution time for traditional model
$Exec_{fl}$	Execution time for federated model

TABLE III: Transaction Cost Analysis

Transaction Costs	
Contract	Gas spent (Wei)
Uploading the hashes over blockchain	26870
Accessing the hashes from blockchain	27513

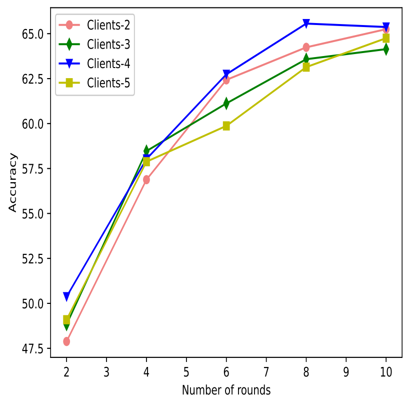


Fig. 7: Accuracy vs. the number of communication federation rounds

TABLE IV: Estimated Values of various parameters

Estimates	
Parameters	Execution Time
T_{FL}	∼100-120 sec (Depends on the model, number of clients, number of communication federation rounds, etc.)
T_{IPFS}	∼5-10 sec
T_{BC}	∼10-12 sec
T_{Others}	∼10-15 sec
T_{Total}	∼125-157sec

D. Transaction Cost Analysis for Smart Contracts

The exchange rates of the present day are, 10^{18} wei $=$ 1 Ether $=$ 2670 USD. As shown in Table III, the gas costs incurred while uploading the model hashes over blockchain and accessing the model hashes from blockchain are very less, which proves that the model is scalable.

E. Latency Analysis

Table IV shows the execution time for different parameters. Here, T_{FL} is the completion time for FL process, T_{IPFS} is the time required to store the parameters over IPFS, T_{BC} is the time spent for the transactions in Ethereum blockchain, and T_{Others} represents various other latencies involved such as encryption-decryption delays.

$$T_{Total} = T_{FL} + T_{IPFS} + T_{BC} + T_{Others}$$

We observe a slight increase in the execution time of our model as compared to the traditionally trained model, as shown in Table I. This is because of the additional latency of various modules such as the conversion of model updates into hashes through IPFS and then the process of storing those hashes over blockchain. The FL process also incurs a slight increase in the training conclusion process, since various tasks are performed during the training process. Our approach also eliminates the server's influence in the training process.

Therefore, the proposed model provides more security and robustness while giving satisfactory performance results.

Moreover, the security mechanisms used in the model ensure that the data does not get compromised at a central server.

V. Conclusion and Future work

In this paper, we proposed a decentralized, secure and transparent mechanism to store and access data. Although our proposed mechanism incurred a slight increase in the training process, it eliminates a single server's influence as well as provides scalability. The integration of federated learning ensures that the data of the users' devices stay on the devices itself. Moreover, the model parameters are saved in an encrypted form over the blockchain, thus providing security against unauthorized access and malicious attacks. Therefore, our model provides a secure manner to share user updates with the manufacturers while maintaining the overall performance of the system.

In this work, we performed the experiments in a simulated environment with a freely available dataset, CIFAR-10, instead of using a real-world IoT dataset. In the future, we plan to implement our approach in a real federated environment with an IoT dataset. Moreover, as the number of users increases, the size of the model parameters and the number of transactions on the blockchain could also increase, potentially leading to scalability issues. Therefore, we will explore ways to make our mechanism more scalable to support a larger number of users.

Acknowledgment

This work was partially supported through a fellowship sponsored by the Tata Consultancy Services (TCS), India.

References

[1] F. Al-Turjman, H. Zahmatkesh, and R. Shahroze, "An overview of security and privacy in smart cities' IoT communications," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, p. e3677, 2022.

[2] 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*, pp. 1273–1282, PMLR, 2017.

[3] V. Agarwal and S. Pal, "Securing IoT with blockchain: Challenges, applications, and techniques," *Securing IoT and Big Data*, pp. 15–38, 2020.

[4] B. Ghimire and D. B. Rawat, "Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things," *IEEE Internet of Things Journal*, 2022.

[5] T. T. A. Dinh, R. Liu, M. Zhang, G. Chen, B. C. Ooi, and J. Wang, "Untangling blockchain: A data processing view of blockchain systems," *IEEE transactions on knowledge and data engineering*, vol. 30, no. 7, pp. 1366–1385, 2018.

[6] G. Zyskind, O. Nathan, *et al.*, "Decentralizing privacy: Using blockchain to protect personal data," in *IEEE Security and Privacy Workshops*, pp. 180–184, 2015.

[7] C. H. Liu, Q. Lin, and S. Wen, "Blockchain-enabled data collection and sharing for industrial IoT with deep reinforcement learning," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3516–3526, 2018.

[8] V. Agarwal and S. Pal, "Blockchain meets IoT: A scalable architecture for security and maintenance," in *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 53–61, IEEE, 2020.

[9] J. Bacon, J. D. Michels, C. Millard, and J. Singh, "Blockchain demystified: a technical and legal introduction to distributed and centralized ledgers," *Rich. JL & Tech.*, vol. 25, p. 1, 2018.

[10] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv preprint arXiv:1712.07557*, 2017.

[11] Z. Yu, J. Hu, G. Min, H. Lu, Z. Zhao, H. Wang, and N. Georgalas, "Federated learning based proactive content caching in edge computing," in *IEEE Global Communications Conference*, pp. 1–6, 2018.

[12] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4177–4186, 2019.

[13] P. C. M. Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe, and M. Atiquzzaman, "A trustworthy privacy preserving framework for machine learning in industrial IoT systems," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6092–6102, 2020.

[14] J. Passerat-Palmbach, T. Farnan, M. McCoy, J. D. Harris, S. T. Manion, H. L. Flannery, and B. Gleim, "Blockchain-orchestrated machine learning for privacy preserving federated learning in electronic health data," in *IEEE International Conference on Blockchain*, pp. 550–555, 2020.

[15] P. Zhang, H. Sun, J. Situ, C. Jiang, and D. Xie, "Federated transfer learning for IIoT devices with low computing power based on blockchain and edge computing," *IEEE Access*, vol. 9, pp. 98630–98638, 2021.

[16] Z. Su, H. Wang, H. Wang, and X. Shi, "A financial data security sharing solution based on blockchain technology and proxy re-encryption technology," in *IEEE 3rd International Conference of Safe Production and Informatization*, pp. 462–465, 2020.

[17] S. Malik, V. Dedeoglu, S. S. Kanhere, and R. Jurdak, "Trustchain: Trust management in blockchain and IoT supported supply chains," in *IEEE International Conference on Blockchain*, pp. 184–193, 2019.

[18] J. Benet, "IPFS-content addressed, versioned, p2p file system," *arXiv preprint arXiv:1407.3561*, 2014.

[19] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, P. P. de Gusmão, and N. D. Lane, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[20] A. Krizhevsky, "Learning multiple layers of features from tiny images," tech. rep., 2009.

[21] A. Gagandeep, P. Kumar, *et al.*, "Analysis of different security attacks in MANETs on protocol stack a-review," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 1, no. 5, pp. 269–75, 2012.