applied sciences

MDPI

Article

Blockchain and Access Control Encryption-Empowered IoT Knowledge Sharing for Cloud-Edge Orchestrated Personalized Privacy-Preserving Federated Learning

Jing Wang [1,2] and Jianhua Li [1,2,*]

[1] School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China; wangjing08@sjtu.edu.cn
[2] Shanghai Key Laboratory of Integrated Administration Technologies for Information Security, Shanghai 200240, China
* Correspondence: lijh888@sjtu.edu.cn

Abstract: Federated learning (FL) is emerging as a powerful paradigm for distributed data mining in the context of Internet of Things (IoT) big data. It addresses privacy concerns associated with data outsourcing by enabling local data training and knowledge (i.e., model) sharing. However, simplistic local knowledge sharing can inadvertently expose user privacy to advanced attacks, such as model inversion or gradient leakage. Furthermore, achieving fine-grained and personalized privacy protection for IoT users remains a challenge. In this paper, we propose a novel solution called hierarchical blockchain-empowered cloud-edge orchestrated federated learning (HBCE-FL) to address these challenges. HBCE-FL is designed to provide secure, intelligent, and distributed data analysis for IoT users. To tackle FL's privacy issues, we develop a multi-level access control encryption and blockchain-based approach for sharing IoT knowledge within the HBCE-FL framework. Our approach classifies IoT users into different levels based on their individual privacy requirements, enabling fine-grained privacy protection. The blockchain is employed for identity authentication, key management, and message sanitization. For scenarios involving IoT users with non-IID data, we integrate federated multi-task learning into HBCE-FL to ensure fairness, robustness, and privacy. Finally, we conduct experiments using classic MNIST and CIFAR10 datasets to validate our approach. The experimental results illustrate that HBCE-FL effectively achieves personalized privacy-preserving FL without losing IoT data availability. Regardless of whether IoT data are homogeneous or heterogeneous, our approach enhances model accuracy and convergence rates by enabling secure IoT knowledge access and sharing for IoT users.

Keywords: federated learning; privacy protection; blockchain; knowledge sharing; Internet of Things

Citation: Wang, J.; Li, J. Blockchain and Access Control Encryption-Empowered IoT Knowledge Sharing for Cloud-Edge Orchestrated Personalized Privacy-Preserving Federated Learning. *Appl. Sci.* **2024**, *14*, 1743. https://doi.org/10.3390/app14051743

Academic Editor: Luis Javier Garcia Villalba

Received: 16 January 2024
Revised: 15 February 2024
Accepted: 18 February 2024
Published: 21 February 2024

1. Introduction

As the Internet of Things (IoT) continues to grow, the proliferation of sensors and smart devices has become substantial. The raw data generated by these IoT devices is extensively distributed among numerous sensors and clients [1], offering the potential for data mining and knowledge discovery [2]. Federated learning (FL), a solution for training data locally and collaboratively building a shared global model, has garnered significant attention [3,4]. Unlike conventional methods, FL clients share knowledge, which consists of model parameters and gradients trained from local data [5], rather than sharing raw data [6]. This approach introduces a novel measure of collaboratively training machine learning models across multiple devices while preserving privacy [7]. Unfortunately, recent research reveals the vulnerability of local training data being reconstructed from shared gradients in FL [8,9], risking the violation of participant privacy during the model updating process [10,11]. Numerous efforts have attempted to integrate privacy protection technologies with FL, primarily employing techniques such as secure multi-party computing

(SMC) [12], homomorphic encryption [13], and differential privacy [14]. SMC and homomorphic encryption techniques all have a high overhead and some complex operations are difficult to handle. Differential privacy, which safeguards data through noise introduction, impacts data availability and leads to a decrease in model accuracy. The cost of different privacy protection mechanisms is reduced accuracy or efficiency [15], and they introduce an inevitable conflict between protecting privacy and achieving higher training performance for clients [16]. Moreover, in distributed collaborative training, it is difficult to guarantee that participants are trustworthy. Untrusted clients participating in training with malicious or erroneous data may compromise the accuracy of the global model and even lead to more serious consequences. Blockchain is a distributed tamper-proof, encrypted linked data block, and some studies introduce blockchain technology into a FL framework to store access control strategies [17] and leave traces of interactive information [18]. However, the transparency of blockchain storage still poses the risk of data leakage.

Diverse device attributes and variations in the importance of local IoT data have introduced a multitude of privacy-preserving requirements among FL clients. High-privacy clients, in particular, are inclined to safeguard their data and knowledge from untrusted servers. Yet, isolating clients based on their privacy needs—separating clients with high privacy requirements from their general counterparts for individual model training—often carries the risk of diminishing model performance due to reduced data sample availability. Our observation leads us to consider information flow control as a viable privacy protection mechanism. In our approach, we categorize both clients and servers according to their security and privacy requirements, devising access control regulations that dictate the FL process between different security tiers. High-security level clients can collaborate by sharing data for training, with data aggregation taking place on high-security level servers. In contrast, servers and clients at lower-security levels are restricted from accessing the data and knowledge of high-security levels clients. The control of information flow is instantiated through the access control encryption (ACE) algorithm [19]. The ACE algorithm administers information flow by introducing a third-party entity called the 'sanitizer' between the sender and receiver to 'sanitize' the encrypted message. The sanitizer, operating without knowledge of the message content and access policy, controls who can read the encrypted message and regulates who can transmit the message by re-encrypting the received message using a special key [20]. Notably, the sanitizer can be served by edge servers within the IoT.

We propose a hierarchical blockchain-empowered cloud-edge orchestrated FL (HBCE-FL) framework in IoT to provide intelligent and distributed data analysis for IoT users. In HBCE-FL, we designed a multi-level access control encryption-based IoT knowledge sharing approach to solving the problem of user privacy leakage in FL. This allows IoT users to participate in FL according to their personal privacy requirements and provides personalized federated learning (PFL) for users of non-IID data to achieve better data analysis results. The main contributions of this paper are as follows:

- We propose an IoT HBCE-FL framework, where knowledge flows securely among local clients, edge blockchain, and cloud servers. This can provide a secure and privacy-preserving data analysis scheme for IoT users.
- We design a multi-level ACE-based IoT knowledge sharing approach in HBCE-FL, which solves the privacy issues in FL by grading security requirements and controlling IoT knowledge flow.
- We introduce blockchain to solve the trust problem of servers and users, improving the reliability of the system. The sanitization of knowledge through smart contracts ensures that knowledge is not tampered with.
- The effectiveness of our framework is validated by security analysis and experimental evaluation. IoT users with different data distributions can all achieve better model accuracy and convergence speed through secure IoT knowledge access and sharing in our designed PFL.

The rest of this paper is organized as follows. Section 2 shows the related work of privacy protection methods in FL. Section 3 describes the overview and main elements of the proposed HBCE-FL architecture. Section 4 illustrates the specific models of the proposed ACE and blockchain-based IoT knowledge sharing scheme. Section 5 describes PFL algorithms with multi-level privacy protection. Security analysis and performance evaluation are provided in Section 6, and Section 7 concludes the paper.

2. Related Work

2.1. Privacy Protection Methods in Federated Learning

With the research on FL gradually developing, the security and privacy protection issues have attracted wide attention. Privacy protection methods in FL are mainly divided into three categories: secure multi-party computation (SMC), differential privacy, and homomorphic encryption.

The key to SMC is zero knowledge, where both parties of communication know nothing but input and output. While this concept is idealized, its practical implementation often necessitates intricate protocols, leading to potential inefficiencies. Xu et al. [21] proposed a scheme to mitigate the impact of unreliable users and protect the privacy of all users' information through secure two-party computation (2PC). Mohassel et al. [22] trained models with two semi-honest servers via the SMC framework, and they also designed a 3PC framework [23] with an honest majority. But when the servers collude together, the security mechanism will be destroyed.

Differential privacy is a privacy-preserving method, which masks the sensitive attributes of data by adding noise. Zhao et al. [24] proposed a scheme, SecProbe, which uses noise to perturb the loss function of deep neural network models. Geyer et al. [25] hid clients' contributions by introducing differential privacy to protect local data in FL. However, adding noise inevitably has an impact on data availability. The solutions using differential privacy often involve a trade-off between utility and privacy [26].

Homomorphic encryption protects data privacy by encrypting the parameters exchanged between clients and servers in FL. This enables the server to calculate the ciphertext of the parameters directly, which can achieve the same effect as the aggregation of plaintext. Phong et al. [27] proposed a scheme to protect model parameter information using additively homomorphic encryption in collaborative deep learning. Park et al. [28] designed an algorithm that enables the server to aggregate the local model parameters which are encrypted using different keys. In practice, evaluating nonlinear functions in FL with additively homomorphic encryption requires polynomial approximation, which also leads to a trade-off between accuracy and privacy [29].

Current privacy protection methods often compromise efficiency or accuracy. However, we offer personalized privacy protection for FL clients through hierarchical access control, representing a groundbreaking design. Clients and servers are categorized into different levels based on privacy requirements and security. The management of knowledge flow among varying security levels is executed through cryptography. Importantly, this solution does not compromise data availability or model accuracy.

2.2. Blockchain in Federated Learning

Blockchain, with its characteristics of being distributed, transparent, autonomous, and immutable, plays a significant role in FL and its application systems according to Ref. [30]. A traditional FL framework typically consists of a single central server and many users (or devices, clients). Blockchain, with its high stability and security, has become a popular technology to solve the single point of failure problem of the centralized server under the FL framework. Li et al. [31] proposed a blockchain-assisted decentralized FL (BLADE-FL) that can solve the single point of failure problems in traditional FL systems but does not deal with privacy issues. Wu et al. [32] designed a blockchain-based FL framework that is decentralized and can operate normally in a state without the master node, with good privacy and robustness. Blockchain technology generates a variety of consensus

mechanisms, making it a viable tool for building a secure collaborative learning mechanism among untrusted users. An FL framework based on the permissioned blockchain is proposed in [33] to address the lack of trust between the end users. In the FL scheme proposed by the author in [34], the blockchain is responsible for computing and storing the reputation of the participating nodes. Blockchain enables meticulous access control through smart contracts, facilitating efficient data operations in the system by mitigating human errors and fraud [35]. Moreover, blockchain serves a critical function in the trusted storage of FL knowledge [36], the sharing and trading of models [37], participant incentives [38], and more. In our solution, blockchain is employed to tackle trust and synchronization issues among distributed edge servers, ensuring the reliability of knowledge sharing in the FL process.

3. Hierarchical Blockchain-Empowered Cloud-Edge Orchestrated FL Architecture

In this section, we describe the threat of data privacy leakage in FL and provide an overview of the proposed HBCE-FL architecture for IoT, introducing its main elements.

3.1. Threat Model

There is a threat of data privacy leakage in the IoT. Although private data are retained in local clients for training in FL, gradient leakage attacks make it possible to analyze gradients to obtain information about local data. Additionally, malicious senders may collude with third parties to leak data or compromise model privacy during the parameter-sharing process. Such privacy leakage cannot be averted simply by using encryption methods, like having the sender embed it in the randomness of the ciphertext, as the malicious receiver knows how to extract it. Therefore, limiting one-way knowledge flows between different local clients with hierarchical levels is essential for achieving a secure system.

It is normal for multiple training tasks to occur simultaneously on multiple servers and clients. Restricting information flow is often needed, especially for controlling who can write. Routers and other edge servers often play the roles of forwarding and filtering information. There are security risks if the edge server is given access to control rights. Edge servers tend to be seen as honest and curious, meaning that they want to learn more about what they receive.

3.2. Overview of Proposed HBCE-FL Architecture

The architecture of HBCE-FL for IoT consists of the local model training layer, the edge blockchain sanitizing layer, and the cloud aggregation layer, as visually represented in Figure 1. The local model training layer consists of IoT edge devices such as smart vehicles, which have a large amount of local private data. These devices employ in-built programs and algorithms to process and train on these data, ultimately generating essential knowledge in the form of FL parameters such as gradients. These devices can establish communication channels with the nearest edge server, enabling parameter exchange and the acquisition of updated model parameters. The edge blockchain sanitizing layer consists of edge servers such as communication base stations, where the knowledge blockchain is deployed. These edge servers play a pivotal role in linking smart devices to cloud servers, assuming responsibility for message sanitization and broadcast. Finally, the cloud aggregation layer includes multiple independent servers, each tailored for specific learning tasks. Servers at different levels serve different clients and do not collaborate or communicate with each other, facilitating data and client privacy isolation. To illustrate this within a practical context, consider a smart home environment, where the local model training layer represents an array of smart devices, including personal computers, wearable devices, and household appliances. In this scenario, routers serve as the servers within the edge blockchain sanitizing layer, while the cloud aggregation layer is the data center under the management of smart device manufacturers and smart home service providers.

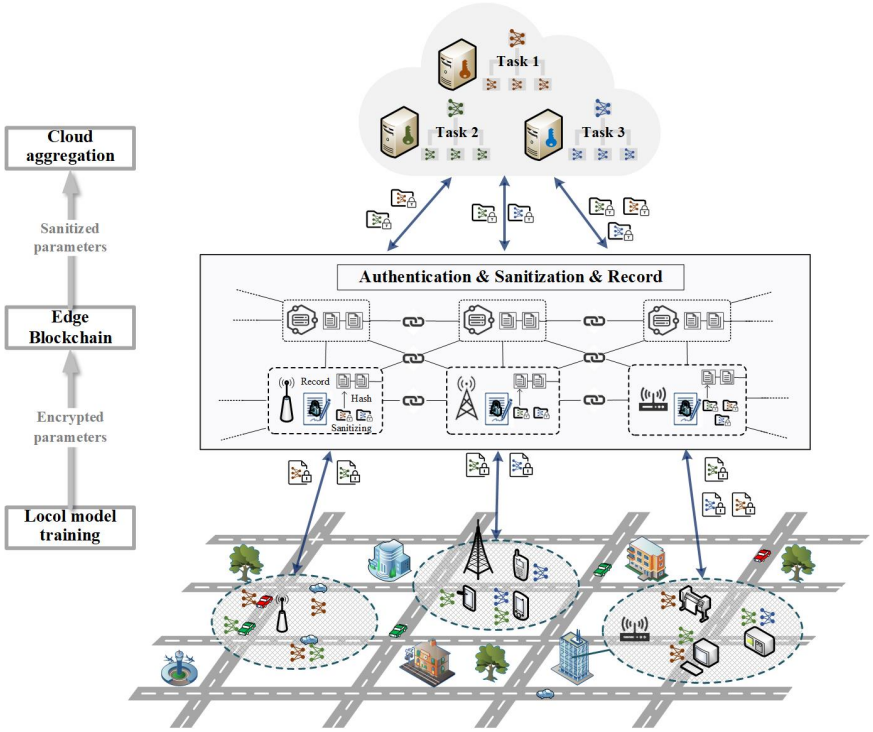


Figure 1. The architecture of our proposed HBCE-FL.

3.3. The Main Elements in HBCE-FL Architecture

The proposed HBCE-FL architecture contains four main elements: trusted authority, local client, edge blockchain, and cloud server.

- Trusted authority: A trusted authority, often controlled by a government department or industry federation, is a fully trusted entity crucial for system initialization. It securely communicates with edge servers via reliable connections. Edge servers and IoT devices must register with this entity to gain system legitimacy. The trusted authority defines HBCE-FL specifics, such as access control policies, user rights, data structures, parameter lifecycles, etc.
- Local client: Any IoT device has the capability to generate data. The local client is responsible for processing and training these data locally, subsequently encrypting the resulting model parameters (knowledge) using its designated key. The encrypted text is delivered to the nearest edge server.
- Cloud server: The cloud aggregation layer, consisting of independent servers with varied security levels, hosts diverse learning tasks. Upon receiving sanitized ciphertext from edge servers, cloud servers decrypt using private keys, aggregating plaintext to update the model.
- Edge blockchain: The edge blockchain layer comprises many edge servers, such as communication base stations, responsible for connecting smart devices to cloud servers and handling message sanitization and forwarding. Identity authentication for FL participants, key distribution, and message sanitization is performed through smart contracts. The edge server is considered a semi-trusted entity, meaning it accurately executes pre-configured algorithms but maintains a level of curiosity about the privacy of the forwarded messages. Due to its interest in the information transmitted within the network and for facilitating subsequent operational analyses, it may anticipate acquiring additional information from the forwarded messages. While it does not inherently engage in activities that compromise the system or user interests, there exists a possibility of exploitation by malicious actors, thereby resulting in the compromise of user privacy.

4. ACE and Blockchain-Based IoT Knowledge Sharing Scheme

4.1. Access Control Model of the IoT Knowledge Sharing

To address the varied privacy requirements of distinct IoT users, clients within the HBCE-FL system can be categorized into different levels of privacy protection. Clients at low-security levels should be restricted from acquiring knowledge from those at higher security levels, mitigating potential privacy concerns arising from model inversion attacks. In light of potential gradient leakage, the model aggregation task for high-security level clients should be entrusted to a more secure cloud server. Simultaneously, high-security level servers should possess the capability to aggregate a larger number of local models, thereby enhancing services for users at higher security levels. Our approach incorporates mandatory access control to describe the accessibility for IoT knowledge flow.

Mandatory access control surpasses discretionary and role-based access control in terms of security. The Bell–La Padula (BLP) model, a classic model in mandatory access control, classifies security levels based on information sensitivity. It enforces one-way information flow through communication rules [39] and ensures data confidentiality with a "no read/no write" principle [40]. Essentially, the BLP model restricts users with lower security levels from reading highly sensitive information and prevents the writing of such information to low-security areas. This aligns seamlessly with the privacy needs of IoT knowledge flow in our scheme.

There are n security levels of senders S_i and receivers R_j in the HBCE-FL system ($i, j \in [1, \ldots, n]$). The predicate $P : [n] \times [n] \to \{0, 1\}$ indicates whether communication is allowed between the sender and the receiver. $P(i, j) = 1$ means that S_i is allowed to send to R_j, while $P(i, j) = 0$ means it is not allowed. The access control policy of IoT knowledge flow can be defined as: $P(i, j) = 1 \leftrightarrow i \geq j$.

For example, in an HBCE-FL system with three security levels, the security levels are classified as Level 1: top-secret; Level 2: secret; and Level 3: public. Level 1 is the highest security level, and then successively lowered. The connectivity of the IoT knowledge flow between the different security levels is shown in Table 1.

Table 1. Access control policy of IoT knowledge sharing.

Role	Level 1 Receiver	Level 2 Receiver	Level 3 Receiver
Level 1 Sender	✓	×	×
Level 2 Sender	✓	✓	×
Level 3 Sender	✓	✓	✓

✓: Communicable. ×: Non-Communicable.

4.2. Multi-Level ACE-Based IoT Knowledge Sharing Algorithm

We implement a multi-level IoT knowledge sharing scheme based on the ACE algorithm [19]. We define ACE^n as an access control encryption algorithm with n security levels. The construction of ACE^1 is as follows:

Setup. With the security parameter λ, this algorithm produces the public parameter $pp = (G, g, q, h)$ and the master secret key $msk = (\alpha, \beta)$, where $\alpha, \beta \in \mathbb{Z}_p$ is randomly selected. The message space $M = G$. The ciphertext spaces for the sender and the sanitizer are C and C', respectively. The details are shown as follows.

- Define a cyclic group $G = \langle g \rangle$ with prime order q.
- Compute $h = g^{\beta}$.

KeyGen. Input master secret key msk, algorithm outputs encryption key ek, decryption key dk, and sanitizer key rk:

$$ek = \alpha, dk = -\beta, rk = -\alpha. \tag{1}$$

Enc. Once input an encryption key ek and a message m, the algorithm selects $s_1, s_2 \in \mathbb{Z}_q$ randomly and computes the ciphertext $c = (c_1, c_2, c_3, c_4) \in C$:

$$c_1 = g^{s_1}, c_2 = g^{s_2}, c_3 = g^{ek} h^{s_1}, c_4 = m h^{s_2}. \tag{2}$$

San. Once a ciphertext $c = (c_1, c_2, c_3, c_4) \in C$ and a sanitizer key rk are input, the algorithm randomly selects $r_1, r_2 \in \mathbb{Z}_q$ and computes the sanitizer ciphertext $c' = (c'_1, c'_2) \in C'$:

$$c'_1 = c_2 c_1^{r_1} g^{r_2}, c'_2 = c_4 (g^{rk} c_3)^{r_1} h^{r_2}. \tag{3}$$

Dec. Once a sanitizer ciphertext $c' = (c'_1, c'_2) \in C'$ and a decryption key dk are input, the algorithm recovers a message:

$$m' = c'_2 (c'_1)^{dk}. \tag{4}$$

Next, we formulate a multi-level IoT knowledge sharing scheme based on ACE[1]. The idea is to execute the ACE[1] algorithm n times and distribute n decryption keys to the receivers of the n security levels. A sender's key comprises a set of the corresponding encryption keys determined by the access policy $P(i, j) = 1$. Each message consists of n parts, encompassing ciphertext and random numbers. The sender encrypts information using all available keys, placing random numbers where encryption keys are unknown. After receiving the message, the sanitizer processes each part of the message with the sanitizer key and broadcasts the sanitized message to all receivers. A receiver decrypts the part corresponding to its security level using its decryption key to obtain the knowledge. Table 2 illustrates the key and ciphertext structure for a three-security-level system. The table also includes decryption results for the ciphertext when the receiver and sender share the same level. R_{12}, R_{13}, R_{23} represent the random ciphertext from C. The algorithm ACE^n is defined as follows:

Table 2. The encryption and decryption process of the ACE[3] with the BLP access control policy.

i	Encryption Key ek_i	Ciphertext c_i	Sanitizer Key rk_i	Decryption Key dk_i	Decryption Result m'_i
1	$\{ek_1\}$	$\{Enc(ek_1, m_1), R_{12}, R_{13}\}$	$\{rk_1, rk_2, rk_3\}$	dk_1	$\{m_1, \perp, \perp\}$
2	$\{ek_1, ek_2\}$	$\{Enc(ek_1, m_2), Enc(ek_2, m_2), R_{23}\}$	$\{rk_1, rk_2, rk_3\}$	dk_2	$\{\perp, m_2, \perp\}$
3	$\{ek_1, ek_2, ek_3\}$	$\{Enc(ek_1, m_3), Enc(ek_2, m_3), Enc(ek_3, m_3)\}$	$\{rk_1, rk_2, rk_3\}$	dk_3	$\{\perp, \perp, m_3\}$

Setup. Given the security parameter λ and access policy $P : [n] \times [n] \to \{0, 1\}$, this algorithm runs n copies of the ACE[1].$Setup$ algorithm and output the public parameter $pp = \{pp_i^1\}_{i \in [1,n]}$. The master secret key $msk = \{ek_i^1, dk_i^1, rk_i^1\}_{i \in [1,n]}$:

$$\begin{aligned} &(pp_i^1, msk_i^1) \leftarrow \text{ACE}^1.Setup(1^\lambda); \\ &ek_i^1 \leftarrow \text{ACE}^1.Gen(msk_i^1, sen); \\ &dk_i^1 \leftarrow \text{ACE}^1.Gen(msk_i^1, rec); \\ &rk_i^1 \leftarrow \text{ACE}^1.Gen(msk_i^1, san). \end{aligned} \tag{5}$$

KeyGen. Once the security level $i \in [1, \ldots n]$, the identity $\{sen, rec, san\}$, and msk are input, the algorithm outputs the three kinds of keys in the system:

$$\begin{aligned} &ek_i = \{ek_j^1\}_{j \in M}, \ where \ M(i) = \{j | P(i, j) = 1\}; \\ &dk_i = dk_i^1; \\ &rk_i = \{rk_j^1\}_{j \in [1,n]}. \end{aligned} \tag{6}$$

Enc. Once the message m and encryption key ek_i are input, the algorithm computes the ciphertext $c = (c_1, \ldots, c_n)$ for $t \in [1, \ldots n]$:

$$c_t = \begin{cases} c_t^1 \leftarrow \text{ACE}^1.Enc(ek_t^1, m), & ek_t^1 \in ek_i; \\ c_t^1 \leftarrow_{\$} C_t^1, & ek_t^1 \notin ek_i. \end{cases} \tag{7}$$

San. Once the ciphertext c and a sanitizer key rk_i are input, the algorithm sanitizes each part of the ciphertexts and outputs the sanitized ciphertext $c' = (c'_1, \ldots, c'_n)$:

$$c'_i \leftarrow \text{ACE}^1.San(rk_i^1, c_i)_{i \in [1,n]}. \tag{8}$$

Dec. Once a sanitized ciphertext c' and a decryption key dk_i are input, the algorithm recovers a message:

$$m' \leftarrow \text{ACE}^1.Dec(dk_i^1, c'_i). \tag{9}$$

In Figure 2, an example of key distribution and knowledge sharing in the ACE^2 system is depicted.

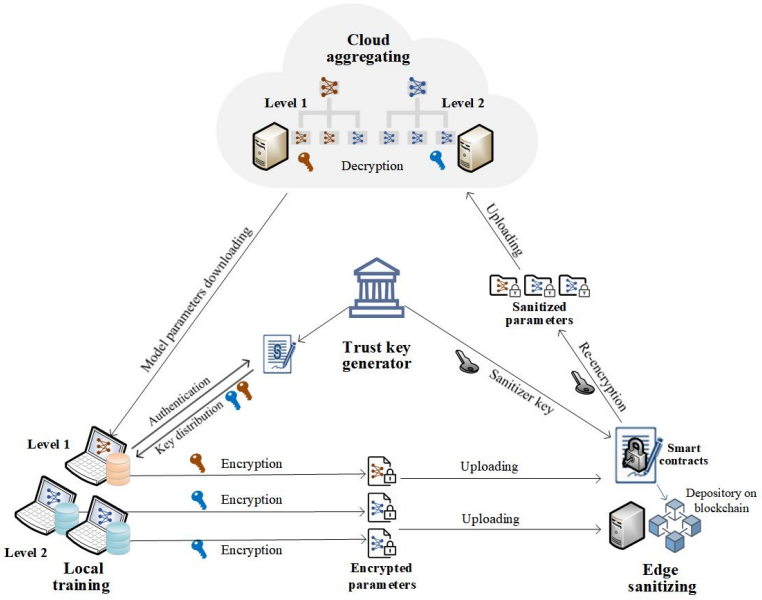


Figure 2. Multi-level ACE and blockchain-based knowledge sharing process.

4.3. Blockchain-Based Authentication and Sanitization Process

The messages from IoT users need to be sanitized and forwarded to the cloud servers via the edge servers. The problems of trust and synchronization between the edge servers can be solved using blockchain technology. A smart contract is a segment of code scripted on the blockchain, and it can be automatically executed without external interference when meeting the set conditions. The authentication of IoT users, key distribution, and the sanitization of messages are all performed by smart contracts in our solution (Algorithm 1), which ensures the correctness and reliability of the service. The messages received by the edge servers are stored in the blockchain for cross-audit and mutual validation, which is shown in Figure 2. All the edge servers maintain a complete data chain jointly, and the data can be easily recovered if a single point of failure occurs.

- **Identity verification and key distribution.** A user who is interested in participating in the FL task sends a request message, including its ID, public key, security level, FL task name, and timestamp, to the nearest edge server. The edge server first deposits the received request message into the blockchain. Then, the authentication is executed. The edge server calculates the hash value of the ID and the public key of requester and compares it with the hash table from the trusted authority to verify the user

identity. This verifies that the security level of the requested task matches the identity permissions after confirming the identity of the requester as a registered and legitimate user. Finally, this verifies that the requested task is within the validity period. After the triple confirmation passes the edge server requests the key packet for this user from the trusted authority. The key packet consists of the task encryption key, and the authentication code of the user identity, task name, and timestamp so that its timeliness and correctness can be verifiable. The key packet is encrypted using the user's public key and the edge server cannot know the exact contents of the key packet. The user receives the key packet and decrypts it using its private key to obtain the requested key.

- **Message sanitization.** The edge server sanitizes and broadcasts all the received messages during FL. Smart contracts start by computing the hash value of these messages, which is then stored on the blockchain for retrospect and comparison. All messages are structurally indistinguishable, preventing any inference about the sender's security level, the intended cloud server receiver's identity, or the specific content of the knowledge. The edge server assesses message timeliness via timestamps, and chooses the sanitizer key based on the task name. This performs the ACE algorithm to sanitize each part of the ciphertext with the sanitizer key if the message is timely. The sanitized message is then broadcast to each cloud server.

Algorithm 1 Identity verification, key distribution, and message sanitization.

Input: Identity ID_i, public key PK_i, security level l_i, task name j, message m, and timestamp ts_1, ts_2, ts_2'.

Output: Key results R and ciphertext M.

Identity verification and key distribution:
$r_{i-j-ts} = (ID_i||PK_i||l_i||j||ts_1)$
Store the request record r_{i-j-ts} on the block
$H_i = H(PK_i||ID_i) = Hash(PK_i||ID_i)$
$ek \leftarrow \perp$ (illegal character)
$r = false$
if $H_i = H_I$ **then**
  if $l_i \in L_i$ and $ts_1 < T_l$ **then**
    if $j \in Tasklist$ and $ts_1 < T_t$ **then**
      Request for $ek_i = Enc_{PK_i}(ek_{ij}||H_i||j||ts_1)$ from the trusted authority
      $ek \leftarrow ek_i$
      $r = true$
    end if
  end if
end if
Message sanitization:
$(c_1||c_2||\ldots||c_n||j||ts_2) \leftarrow m$
if $j \in Tasklist$ and $ts_2 < T$ **then**
  $rk_{ij} \leftarrow rklist$
  for i = 1, 2, . . . , n **do**
    $c_i' \leftarrow \mathrm{ACE}^1.San(rk_{ij}^1, c_i)$
  end for
  $M \leftarrow (c_1'||c_2'||\ldots||c_n'||j||ts_2')$
else
  $M \leftarrow \perp$ (illegal character)
end if
return $R = (ek, r), M$

5. Personalized FL with Multi-Level Privacy-Preserving IoT Knowledge Sharing

5.1. Multi-Level ACE-Based FL Process

In an HBCE-FL system with multi-level ACE-based IoT knowledge sharing, a low-security-level server cannot communicate with high-security-level clients, while high-security-level servers can communicate with clients of the same or lower level. In other words, high-security-level servers can aggregate more local models to deliver better and more secure services to high-level users.

The objective of FL is to train a single global model that minimizes the aggregate loss across all participating clients. Suppose there are K clients in the system, then the aim of FL is to solve:

$$\min_{\theta} G(F^1(\theta), \ldots, F^K(\theta)), \tag{10}$$

where θ is the model parameter that is being optimized across all clients. Each client optimizes its local objective function with respect to the same global parameter θ. Each client has a local data distribution $D^k = P^k(x, y)$. The kth client has the optimization target:

$$\min_{\theta} F^k(\theta) \triangleq E_{(x^k, y^k) \sim D^k}[l(f(x^k; \theta), y^k)], \tag{11}$$

where f is the prediction function, and l is the loss function. We use FedAvg [41], a standard FL aggregation method, that is:

$$\min_{\theta} \sum_{k \in K} \frac{n_k}{\sum_k n_k} F^k(\theta), \tag{12}$$

where n_k is the number of samples on the kth client. In each round of FedAvg, every client updates the model on its local dataset D^k using the stochastic gradient descent rule. $W(j) = \{i | P(i, j) = 1\}$ represents the set of clients i capable of sending messages to server j. The Algorithms 2 and 3 outline the local training and cloud aggregation processes of the proposed multi-level ACE-based privacy-preserving FL. The client computes the local average loss function with the initialized model parameters (lines 2–4 in Algorithm 2) and locally takes one step of gradient descent on the current model (line 5 in Algorithm 2). It then executes the ACE^n.*Enc* the algorithm to encrypt the new model parameter (lines 8–9 in Algorithm 2) to send to the server. The server executes the ACE^n.*Dec* algorithm to decrypt the messages received from the clients containing FL parameter information (lines 1–2 in Algorithm 3). It then selects the client model parameters that comply with the access control rules and engages in the current aggregation process (lines 3–7 in Algorithm 3). Averaging the model parameters produces new parameters for distribution (line 9 in Algorithm 3).

Algorithm 2 Client procedure.

Input: Local data D^k, batch size B, number of local epochs E, initialized model parameters $\theta_{s,t}$, and client encryption key ek.
Output: Ciphertext C.

1: $\theta^k_{s,t} \leftarrow \theta_{s,t}$
2: **for** local epoch $e = 1, 2, \ldots, E$ **do**
3: **for** each batch$\{(x^k_i, y^k_i)\}_{i=1}^{B}$ sampled from D^k **do**
4: $\mathcal{L}(\theta^k_{s,t}, D^k) = \frac{1}{B} \sum_{i=1}^{B} l(\theta^k_{s,t}, x^k_i, y^k_i)$
5: $\theta^k_{s,t} \leftarrow \theta^k_{s,t} - \eta \nabla \mathcal{L}(\theta^k_{s,t}, D^k)$
6: **end for**
7: **end for**
8: $m \leftarrow \theta^k_{s,t}$
9: $C \leftarrow \text{ACE}^n.Enc(ek, m)$
10: **return** C

Algorithm 3 Server procedure.

Input: Maximum number of communication rounds T, client selection ratio Q, sanitized ciphertext C', and server decryption key dk.
Output: The final aggregated global model $\theta_{s,T+1}$

for global round $t = 0, 1, 2, \ldots, T$ **do**
  $m' \leftarrow \mathrm{ACE}^n.Dec(dk, C')$
  for $k \in W$ **do**
    $\theta^k_{s,t} \leftarrow m'$
    $S_t \leftarrow$ sample $\max(Q \cdot k, 1)$ clients
    for $k \in S_t$ **do**
      $C \leftarrow$ **Client Procedure**$(k, \theta'_{s,t})$
    end for
    $\theta_{s,t+1} \leftarrow \sum_{k \in S_t} \frac{1}{|S_t|} \theta^k_{s,t}$
  end for
end for
return $\theta_{s,T+1}$

5.2. Personalized FL of IoT Users with Non-IID Data

The data of clients requiring high privacy protection and general client data do not meet independently and, in some cases, are identically distributed. For example, in the smart home scenario, the privacy protection of health data recorded by wearable smart devices of family members has high requirements. The distribution of these health data and the indoor environment data (like temperature, humidity, etc.) recorded by other smart devices is highly likely to differ. Therefore, the PFL in IoT users with non-IID data also needs to be considered.

In the case of non-IID, different clients adapt different models. We implement personalization through a federated multi-task learning framework, Ditto [42]. There are two "tasks" in the system that need to be considered: the global objective G and the local objective $F^k(\theta)$. To better adapt to the heterogeneity of data between clients, personalized and client-specific models $F^k(\theta)$ need to be emphasized. A regularization term is added to relate the two tasks by bringing each local personalized model close to the optimal global model. The kth client has the bi-level optimization target:

$$\begin{aligned} \min_{\theta} \quad & h_k(v^k; \bar{\theta}) := F^k(v^k) + \tfrac{\lambda}{2} \left\| v^k - \bar{\theta} \right\| \\ s.t. \quad & \bar{\theta} \in \arg\min_{\theta} G(F^1(\theta), \ldots, F^K(\theta)) \end{aligned}, \tag{13}$$

where the hyperparameter λ leads to a trade-off between local and global models. The smaller λ encourages training to approach a model to local data. When λ is set to 0, Ditto is reduced to local models. As λ grows larger, the impact caused by data from other clients continues to increase. When $(\lambda \to +\infty)$, it recovers the global model objective G.

There are three major constraints, namely fairness, robustness, and privacy in FL. The general approach makes it difficult to satisfy all three constraints simultaneously. The heterogeneity of the data distribution is the fundamental reason that limiting fairness and robustness cannot rise simultaneously. PFL helps improve fairness and robustness by learning the different models of each client to adapt to heterogeneity. Privacy can be guaranteed through grading the security requirements and control of the knowledge flow by the ACE algorithm. Therefore, our multi-level privacy-preserving PFL scheme has a good performance in terms of fairness, robustness, and privacy.

Algorithm 4 depicts the local training of PFL in IoT users with non-IID data. It should be noted that the local personalized model that was trained is an objective function including the regularization term, while the update parameters uploaded to the server are calculated without regularization terms. The aggregation process in cloud server is the same as in the IID case, referring to Algorithm 3 for details.

Algorithm 4 Client procedure for personalized FL.

Input: Local data D^k, batch size B, number of local epochs E, initialized model parameters $\theta_{s,t}$, client-specific parameter v^k, interpolation control parameters λ, and client encryption key ek.

Output: Ciphertext C.

$\theta^k_{s,t} \leftarrow \theta_{s,t}$
for local epoch $e = 1, 2, \ldots, E$ **do**
  for each batch$\{(x^k_i, y^k_i)\}_{i=1}^{B}$ sampled from D^k **do**
    $\mathcal{L}(\theta^k_{s,t}, D^k) = \frac{1}{B}\sum_{i=1}^{B} l(\theta^k_{s,t}, x^k_i, y^k_i)$
    $\theta^k_{s,t} \leftarrow \theta^k_{s,t} - \eta\nabla\mathcal{L}(\theta^k_{s,t}, D^k)$
    $v^k = v^k - \eta(\nabla F^k(v^k) + \lambda(v^k - \theta^k_{s,t}))$
  end for
end for
$m \leftarrow \theta^k_{s,t}$
$C \leftarrow \mathrm{ACE}^n.Enc(ek, m)$
return C

6. Security Analysis and Performance Evaluation

We conducted security analysis and experimental evaluation on the key components of the HBCE-FL framework, particularly in terms of privacy preservation, computation time, and model accuracy. Section 6.1 shows the security analysis of the access control rules implemented using ACE and the blockchain deployed on the edge servers. The performance evaluation of the multi-level ACE-based IoT knowledge sharing algorithm is presented in Section 6.2. The experimental evaluation of privacy-preserving PFL is provided in Section 6.3.

6.1. Security Analysis

The multi-level ACE algorithm in our system adheres to the "no read" and "no write" security rules of mandatory access control, ensuring the confidentiality and accessibility of the IoT knowledge flow.

- **Confidentiality:** In our system, the IoT knowledge transfer is encrypted using ACE, instantiated based on the ElGamal public-key encryption scheme [43], which is indistinguishable against selectively chosen plaintext attacks (IND-CPA) and secure under the DDH assumption. In the sanitization phase, the edge server uses the sanitizer key to re-encrypt the received ciphertext, without having knowledge of its specific content. Importantly, the identity of the sender and intended receiver remains undisclosed, as all messages share a consistent format and are broadcast to every cloud server.
- **No read:** All receivers R_j satisfying $P(i, j) = 0$ are unable to access any information about m_i from sender S_i. This fundamental requirement ensures system confidentiality and can be fulfilled using a standard encryption scheme. Mostly, a receiver only decrypts the part of the message corresponding to its identity level for efficiency. When $P(i, j) = 0$, ek_i and dk_j are mismatched. As the receiver's key cannot correctly decrypt information encrypted by the non-corresponding key, obtaining plaintext is impossible, even if the receiver attempts to decrypt all parts of the message.
- **No write:** If $P(i, j) = 0$, no sender S_i can convey information to any receiver R_j. The sanitizer is unable to extract the identity-level information of the sender from C because each part of the ciphertext in message C shares a uniform structure, rendering it indistinguishable. Additionally, the sanitizer employs a specific random algorithm for received message C and cannot comprehend the message's content without the decryption key, preventing it from identifying the sender and receiver. So, a corrupt sender cannot collude with the sanitizer to transmit messages to an unauthorized receiver. On the other hand, if sender S_i seeks to convey information, it may substitute the information for random numbers in the message part corresponding to the receiver.

However, with $P(i, j) = 0$, sender S_i lacks the encryption key corresponding to receiver R_j. As a result, the receiver cannot decrypt the message part with its key after the sanitizer's processing, preventing the receiver from obtaining any information from an unauthorized sender.

On the one hand, the FL model using ACE to set multiple security levels can provide personalized privacy protection for users, and the confidentiality of knowledge is provided by cryptography. On the other hand, blockchain ensures system reliability and that it is tamper-proof.

- **Tamper-proof:** The process of knowledge sanitization is carried out through the execution of the script file within the smart contracts. As long as the smart contract works properly in Ethereum, the re-encryption of the message is executed correctly. The sanitization results will be publicly and permanently saved. The user's messages are checked using the hash function, and the message record is stored on the blockchain. Due to the transparency of blockchain, the tampered messages is traceable and cannot be denied. Each blockchain node in the system can detect any alterations. Consequently, our system can become tamper-proof and achieve the traceability of IoT knowledge.
- **Reliability:** The decentralized feature of the blockchain can mitigate risks associated with a centralized structure. Each edge server holds the data for the entire blockchain, which avoids data recovery problems after a single server failure. Blockchain provides a trust mechanism for the IoT that does not rely on third parties, and its consensus mechanism can solve the problem of trust and synchronization among numerous edge servers. The trust problem of IoT users is solved through registration and authentication process. Our solution guarantees that users receive a reliable and accurate service.

6.2. Performance of the Multi-Level ACE-Based IoT Knowledge Sharing Algorithm

We implement the multi-level ACE algorithm with Java to evaluate the practical performance of the IoT knowledge sharing approach we proposed in HBCE-FL. The communication process between local clients, edge servers, and cloud servers is essentially the process of encrypting, sanitizing, and decrypting transmitted messages. The program is executed on a Windows 10 64-bit operating system, utilizing an Intel(R) Core(TM) i7-8565U CPU clocked at 1.80 GHz and 16 GB of RAM. The cyclic group G of the ACE algorithm is implemented using elliptic curves constructed over a 128-bit finite field, based on an elliptic curve of Type-A. This implementation is facilitated by the JPBC framework, a Java library that ports the Pairing-Based Cryptography (PBC) library. The size of the element in the $\mathbb{Z}_p$ is 20 bits. We did not consider communication delay in order to focus on the running time of the ACE algorithms.

We employ ACE[1] to assess the algorithm's performance. The runtime of each algorithm component, using a 1024-bit plaintext as input, is detailed in Table 3. The time overhead of **Setup** primarily involves initializing the cyclic group G, $\mathbb{Z}_p$, and a primary exponential operation. **Enc** and **San** exhibit similar time overheads, given their identical number of exponential operations and a comparable number of multiplications. **Dec** features a straightforward structure with only one exponential operation. The **KeyGen** performs solely assignment operations, rendering its runtime negligible.

Table 3. Running time of ACE[1] algorithm.

Algorithm	Setup	Enc	San	Dec
Time (ms)	10.97	46.02	47.81	9.66

The performance of ACE algorithms is influenced by the number of clients and the length of the plaintext. We consider the number of clients and plaintext length as variables

to assess the performance of our ACE algorithm. Figure 3 illustrates the time and space overhead of each part of the ACE algorithm. Specifically, Figure 3a,d depict the relationship between the time and space overhead of each algorithmic component and the plaintext length when there is only one client in the system. It can be seen from Figure 3a that the knowledge encryption, sanitization, and decryption time all exhibit a linear increase with the growth of the length of the input knowledge plaintext, albeit the change is minimal. The computation time for each part of the algorithm is essentially in milliseconds, which is negligible for clients and servers capable of performing FL training tasks. Figure 3d indicates that the length of the two ciphertexts is linearly correlated with the length of the input plaintext.

In practical scenarios, we implement a simulation of the HBCE-FL framework with three security levels. The distribution of users across security levels follows a ratio of 1:4:5, with fewer users at higher security levels. Each message has a plaintext length of 1024 bits. The client encrypts a message, which is sanitized and sent to all cloud servers by the edge sanitizer. Consequently, the client executes $\mathrm{ACE}^n.Enc$ only once for each communication round, and the number of clients in the system does not impact an individual client's encryption time. For clients at different security levels, lower-level clients possess more encryption keys, resulting in more encryption operations. Conversely, higher-level clients experience shorter encryption times. As depicted in Figure 3b, the client's level exhibits a linear relationship with encryption time. The encryption time is within the millisecond range, rendering it negligible compared to the local training time.

Figure 3e illustrates how the total edge sanitization time and the decryption time of a single cloud server vary with the number of clients in the system. With each additional client, the sanitizer processes one more message, and the server decrypts an additional message. Both the total sanitization time in the edge sanitizer and the decryption time of a single cloud server exhibit a linear relationship with the number of clients. As the number of users increases, the decryption and sanitization times also increase proportionally. The total sanitization time in the edge is the cumulative time taken by all edge servers to run $\mathrm{ACE}^n.San$ algorithm. In practical scenarios, many edge sanitizers are deployed, with each being responsible for sanitizing messages from a few clients in its vicinity. The decryption burden on a single sanitizer is relatively low. Both sanitization and decryption operations are performed by servers with robust computing capabilities, making milliseconds of computational overhead negligible.

Finally, we observe the impact of the number of security levels in the system on the time and storage overhead of each partial algorithm for a certain amount of clients. there are a total of 30 clients in the system, and the number of security levels is 2, 3, 4, 5, respectively. Each group of experiments follows the principle of having a higher safety level and thus a lower number of clients. As shown in Figure 3c, the time of total sanitization in edge increases with the number of security levels; however, the time of decryption of a single cloud server is not affected. The length of the local ciphertext and sanitized ciphertext increases with the number of security levels in Figure 3f. Because there are more security levels, the more there are ciphertext C and C' components, the more parts need to be sanitized for each edge server. But the decrypted part is certain, which is only related to the total number of clients.

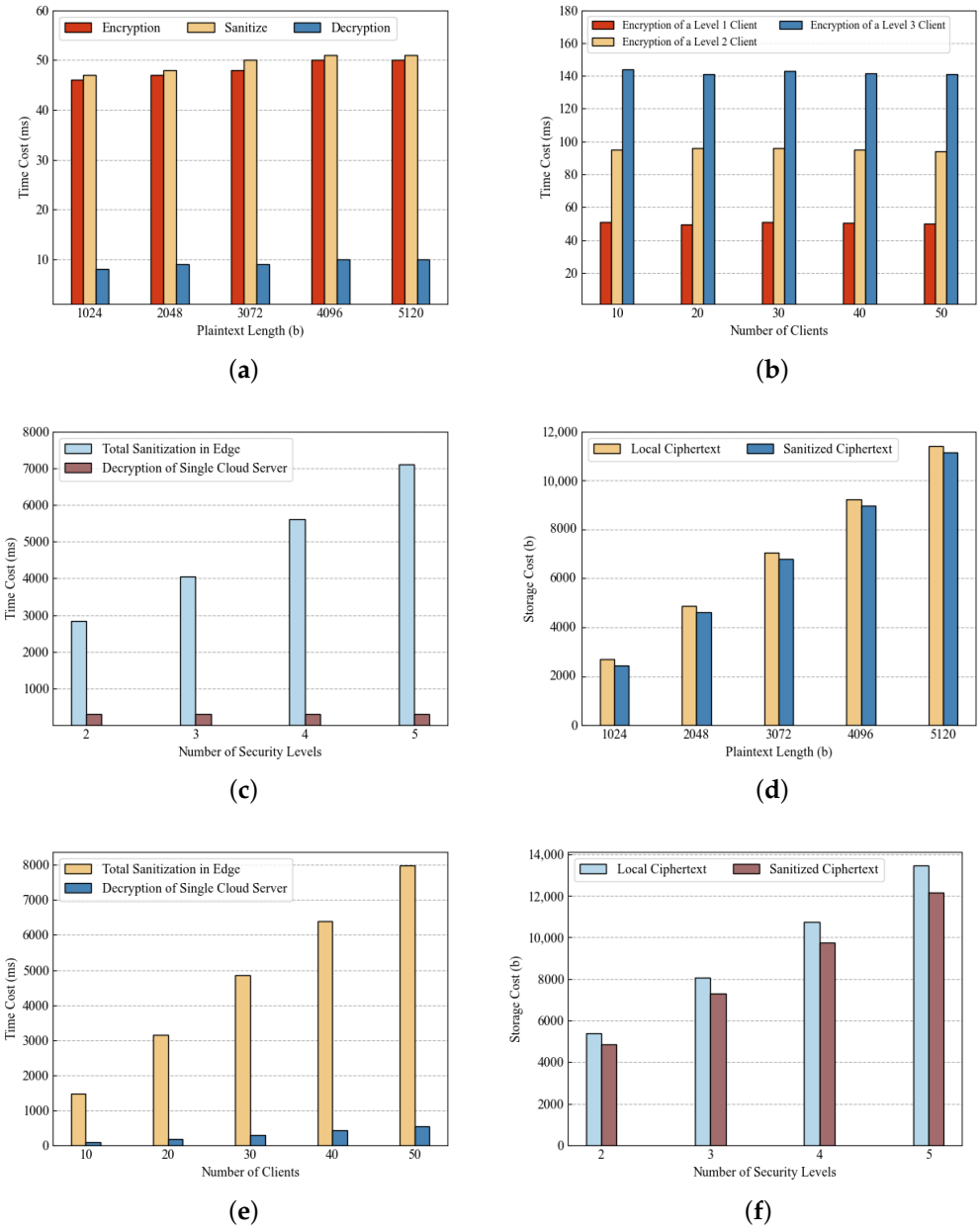


Figure 3. Computational time and storage cost of the ACE algorithm. (**a**) Time cost with different sizes of plaintext length of ACE[1]. (**b**) Encryption time cost with different numbers of clients of ACE[3]. (**c**) Time cost with different numbers of security levels. (**d**) Storage cost with different sizes of plaintext length of ACE[1]. (**e**) Sanitization and decryption time cost with different numbers of clients of ACE[1]. (**f**) Storage cost with different numbers of security levels.

6.3. Performance of Personalized FL with Multi-Level ACE-Based IoT Knowledge Sharing

The proposed multi-level ACE-based personalized FL framework undergoes evaluation on two widely used datasets, MNIST and CIFAR-10, which were both designed for image classification tasks encompassing 10 classes. The training sets are divided among 30 clients in accordance with the IID setting. Under this setup, each client is randomly assigned a uniform distribution across the 10 classes. These 30 clients are further divided into two groups: 5 clients at the secret level and the remaining 25 clients at the public level. We adopted a two-layer CNN with a 5×5 convolution kernel as the backbone model. The training configuration involves 1 local epoch and 500 communication rounds. We utilize the SGD optimizer with a learning rate of 0.005, and the batch size is set to 64.

The training loss curve and test accuracy curve on MNIST are illustrated in Figure 4a,b. For clients at the secret level, FedAvg with multi-level ACE-based IoT knowledge sharing converges to 98.83%, whereas FedAvg without the access control scheme converges to 96.69%. Our solution achieves personalized privacy-preserving federated learning without losing IoT data availability. Notably, our solution converges faster on the clients' decentralized data, as evident from the loss curve. The performance disparity between multi-level ACE-empowered FL and no-ACE FL is more pronounced on CIFAR-10. As depicted in Figure 4c,d, multi-level ACE-empowered FedAvg attains an accuracy of 68.59%, while FedAvg without the access control scheme achieves 51.09%, representing a noteworthy improvement of 17.5%. These experimental results underscore that users can achieve enhanced FL model accuracy and faster convergence rates through our proposed secure IoT knowledge access and sharing approach.

Considering the case that the data of different IoT users are non-IID, a further experimental setup is performed as follows. We simulate the case of non-IID in a distribution-based label imbalance [44] way. The specific method is to assign the proportion of each label sample based on the Dirichlet distribution, which is a commonly used prior distribution for simulating real data distribution. We sample $p_k \sim Dir_N(\beta)$ and allocate a pk, l proportion of the instances of class k to batch l [45]. $Dir(\beta)$ denotes the Dirichlet distribution and concentration parameter $\beta = 0.5$ acquiescently.

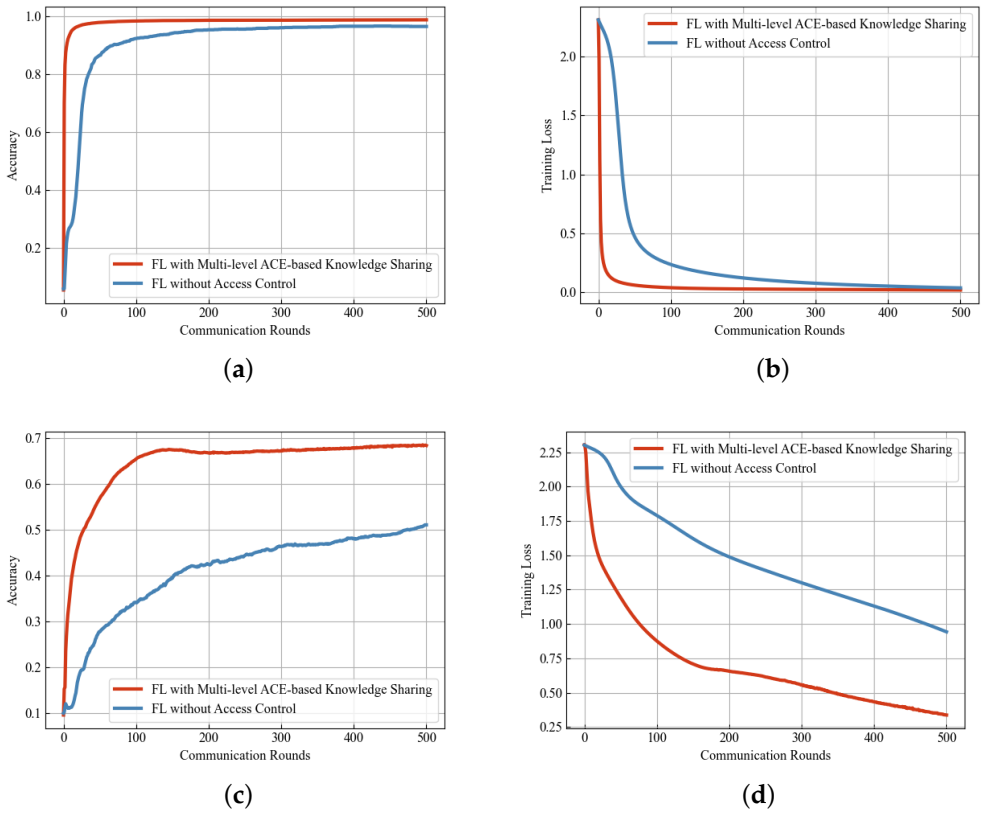


Figure 4. *Cont.*

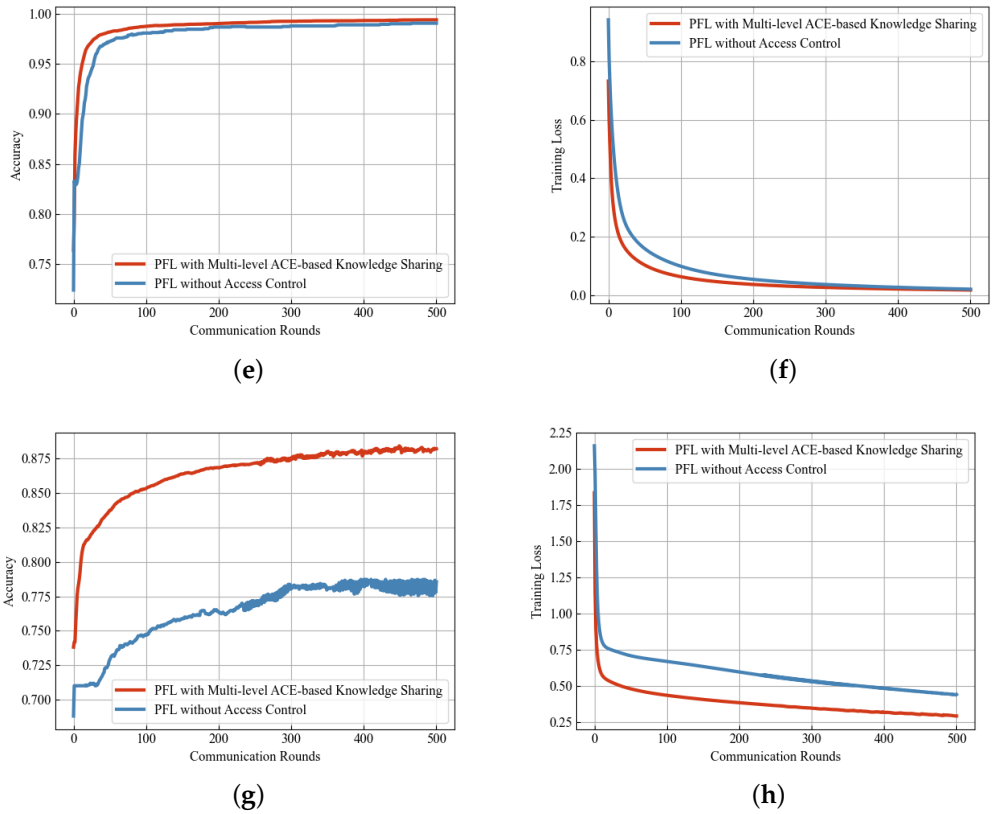
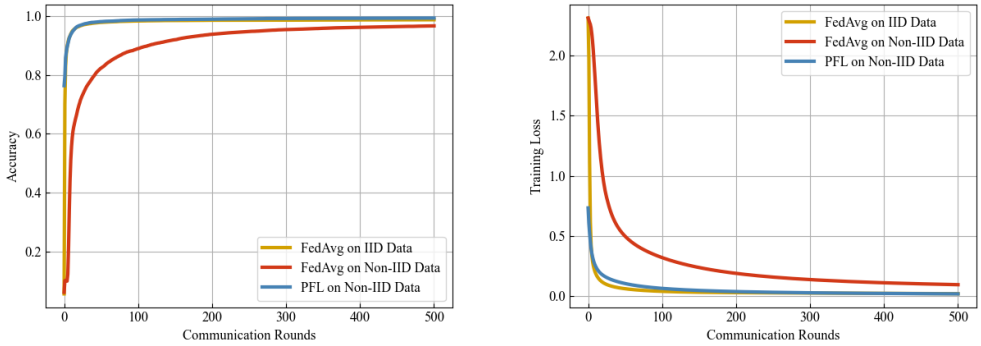


Figure 4. Performance of the proposed privacy-preserving PFL. (**a**) Average accuracy on MNIST with IID data. (**b**) Training loss on MNIST with IID data. (**c**) Average accuracy on CIFAR-10 with IID data. (**d**) Training loss on CIFAR-10 with IID data. (**e**) Average accuracy on MNIST with non-IID data. (**f**) Training loss on MNIST with non-IID data. (**g**) Average accuracy on CIFAR-10 with non-IID data. (**h**) Training loss on CIFAR-10 with non-IID data.

In the non-IID scenario, FedAvg with multi-level ACE-based IoT knowledge sharing achieves an average accuracy of 96.74% on MNIST for secret-level clients, slightly below the IID case (98.83%). This closely aligns with the average accuracy of FedAvg without the access control scheme in the IID scenario (96.69%). Furthermore, both the accuracy and loss function convergence on non-IID data using the FedAvg algorithm are inferior to those on IID data, as depicted in Figure 5. This suboptimal training effect is deemed unacceptable. To enhance knowledge-sharing performance, opting for more suitable personalized federated learning algorithms becomes imperative when dealing with non-IID data from IoT users.

We conduct experiments on MNIST using the proposed PFL algorithm with multi-level ACE-based IoT knowledge sharing. The hyperparameter λ in PFL is set to 0.1, and the other training parameters are set at the same as in the IID case. The average accuracy of the PFL algorithm converged to 99.51% in the setting of the non-IID. Its average accuracy and the convergence rate of the loss function are close to the FedAvg with multi-level ACE-based IoT knowledge sharing on IID data. This result is satisfactory. PFL adapts to the heterogeneity of data by learning the different models trained by each client, which focuses more on the local data distribution. So, it has a better training effect than FedAvg when the data are non-IID. Moreover, the correlation degree of the local model with the global model can be adjusted by varying the value of the hyperparameter λ, so as to achieve a more satisfactory local training accuracy. The performance comparison of FedAvg and PFL algorithms trained on IID data and non-IID data are shown in Figure 5.



(**a**) (**b**)

Figure 5. Performance of the proposed different privacy-preserving FL on different data distributions. (**a**) Average accuracy; (**b**) Training loss.

In the setting of non-IID, we implemented the comparison experiments of the proposed PFL with multi-level ACE-based IoT knowledge sharing and the PFL without the access control scheme. The training loss curve and test accuracy curve on MNIST are shown in Figure 4e,f. For clients of the secret level, the average accuracy of PFL with multi-level ACE-based IoT knowledge sharing converges to 99.51%, which is slightly better than the PFL without an access control scheme (99.06%). It can be observed that our solution converges faster on both the accuracy and the loss curve. We also implement the PFL experiment on CIFAR-10 and the effect is more pronounced. It is shown in Figure 4g,h that the average accuracy of multi-level ACE-empowered PFL converges to 88.21% while PFL without access control scheme achieves 78.55%, with an improvement of 9.66%. Furthermore, the multi-level ACE-empowered PFL performs better in both the average accuracy and the loss convergence rate. The experimental results reveal that the multi-level ACE-based IoT knowledge sharing approach enables IoT users with high privacy protection levels to cooperate with other users for distributed data mining. Users can obtain a better model accuracy and convergence rate through secure IoT knowledge access and sharing regardless of whether their data are IID or non-IID.

7. Conclusions

In this paper, we propose a hierarchical blockchain-empowered cloud-edge orchestrated FL (HBCE-FL) architecture aimed at providing a secure and privacy-preserving data analysis solution for IoT users. In HBCE-FL, we design a multi-level ACE-based IoT knowledge sharing approach to address the challenge of user privacy leakage in FL. This scheme enables IoT users to engage in personalized FL based on their unique privacy requirements. Clients and servers in FL are classified into different levels based on privacy requirements and security. The ACE algorithm regulates the flow of knowledge between the users of varying levels and the cloud server, thereby enforcing access control rules. And, blockchain ensures trust and reliability in services among edge servers. The simulation results of the HBCE-FL framework with three security levels show that the multi-level ACE-based IoT knowledge sharing approach enables IoT users with high privacy protection levels to cooperate with other users for distributed data mining. Our framework achieves multi-level privacy-preserving FL with acceptable computation times.

The proposed multi-level ACE-based personalized privacy-preserving FL framework undergoes evaluation on two widely used datasets, MNIST and CIFAR-10. The average accuracy of our privacy-preserving FL scheme surpasses that of FedAvg without privacy protection by 2.14% and 17.5% on the two datasets, respectively. Additionally, we designed personalized FL for IoT users with non-IID data by having different clients adopt different models. In the non-IID scenario, our scheme achieves average accuracies of 99.51% and 88.21% on the two datasets, respectively, with faster convergence rates. Regardless of

handling homogeneous or heterogeneous data, IoT users can improve FL performance through secure knowledge access and sharing.

Our work exclusively focuses on addressing the FL challenge of data security sharing and privacy protection in the context of determined user privacy requirements. In the future, we will design more meticulous and efficient access control rules and ACE algorithms to adapt to the dynamic changes in user privacy requirements. Additionally, we will delve into strategies for handling unreliable or malicious participants.

Author Contributions: Conceptualization, J.W. and J.L.; methodology, J.W.; simulation and analysis, J.W.; writing—review and editing, J.W. and J.L.; supervision, J.L.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant U20B2048, Grant U2003206, and Shanghai Engineering Research Center of Cyber and Information Security Evaluation (The Third Research Institute of Ministry of Public Security) KFKT2023-006.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author. The data are not publicly available due to privacy and ethical concerns.

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

References

1. Xiao, K.; Gao, Z.; Shi, W.; Qiu, X.; Yang, Y.; Rui, L. EdgeABC: An architecture for task offloading and resource allocation in the Internet of Things. *Future Gener. Comput. Syst.* **2020**, *107*, 498–508. [CrossRef]
2. Lin, X.; Wu, J.; Li, J.; Zheng, X.; Li, G. Friend-as-Learner: Socially-Driven Trustworthy and Efficient Wireless Federated Edge Learning. *IEEE Trans. Mob. Comput.* **2023**, *22*, 269–283. [CrossRef]
3. Yang, Q.; Liu, Y.; Chen, T.; Tong, Y. Federated Machine Learning: Concept and Applications. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–19. [CrossRef]
4. Li, T.; Sahu, A.K.; Talwalkar, A.; Smith, V. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Process. Mag.* **2020**, *37*, 50–60. [CrossRef]
5. Li, G.; Dong, M.; Yang, L.T.; Ota, K.; Wu, J.; Li, J. Preserving Edge Knowledge Sharing Among IoT Services: A Blockchain-Based Approach. *IEEE Trans. Emerg. Top. Comput. Intell.* **2020**, *4*, 653–665. [CrossRef]
6. Gao, Z.; Yang, Y.; Zhao, C.; Mo, Z. CFedPer: Clustered Federated Learning with Two-Stages Optimization for Personalization. In Proceedings of the 2022 18th International Conference on Mobility, Sensing and Networking (MSN), Guangzhou, China, 14–16 December 2022; pp. 171–177. [CrossRef]
7. Zhu, L.; Liu, Z.; Han, S. Deep Leakage from Gradients. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2019; Volume 32.
8. Zhao, B.; Mopuri, K.R.; Bilen, H. iDLG: Improved Deep Leakage from Gradients. *arXiv* **2020**, arXiv:2001.02610.
9. Wei, W.; Liu, L. Gradient Leakage Attack Resilient Deep Learning. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 303–316. [CrossRef]
10. Shokri, R.; Stronati, M.; Song, C.; Shmatikov, V. Membership Inference Attacks Against Machine Learning Models. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 3–18. [CrossRef]
11. Melis, L.; Song, C.; De Cristofaro, E.; Shmatikov, V. Exploiting Unintended Feature Leakage in Collaborative Learning. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2019; pp. 691–706. [CrossRef]
12. Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; McMahan, H.B.; Patel, S.; Ramage, D.; Segal, A.; Seth, K. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In Proceedings of the CCS '17, 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 1175–1191. [CrossRef]
13. Hardy, S.; Henecka, W.; Ivey-Law, H.; Nock, R.; Patrini, G.; Smith, G.; Thorne, B. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv* **2017**, arXiv:1711.10677.
14. McMahan, H.B.; Ramage, D.; Talwar, K.; Zhang, L. Learning Differentially Private Recurrent Language Models. *arXiv* **2017**, arXiv:1710.06963.
15. Bagdasaryan, E.; Poursaeed, O.; Shmatikov, V. Differential Privacy Has Disparate Impact on Model Accuracy. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2019; Volume 32.
16. Yu, T.; Bagdasaryan, E.; Shmatikov, V. Salvaging Federated Learning by Local Adaptation. *arXiv* **2020**, arXiv:2002.04758.

17. Di Francesco Maesa, D.; Mori, P.; Ricci, L. Blockchain Based Access Control. In *Distributed Applications and Interoperable Systems, Proceedings of the 17th International Federated Conference on Distributed Computing Techniques, DisCoTec 2022, Lucca, Italy, 13–17 June 2022*; Chen, L.Y., Reiser, H.P., Eds.; Springer: Cham, Switzerland, 2017; pp. 206–220.
18. Majeed, U.; Hong, C.S. FLchain: Federated Learning via MEC-enabled Blockchain Network. In Proceedings of the 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), Matsue, Japan, 18–20 September 2019; pp. 1–4. [CrossRef]
19. Damgård, I.; Haagh, H.; Orlandi, C. Access Control Encryption: Enforcing Information Flow with Cryptography. In *Theory of Cryptography*; Hirt, M., Smith, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2016; pp. 547–576.
20. Tan, G.; Zhang, R.; Ma, H.; Tao, Y. Access Control Encryption Based on LWE. In Proceedings of the APKC '17, 4th ACM International Workshop on ASIA Public-Key Cryptography, Abu Dhabi, United Arab Emirates, 2 April 2017; pp. 43–50. [CrossRef]
21. Xu, G.; Li, H.; Zhang, Y.; Xu, S.; Ning, J.; Deng, R.H. Privacy-Preserving Federated Deep Learning With Irregular Users. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 1364–1381. [CrossRef]
22. Mohassel, P.; Zhang, Y. SecureML: A System for Scalable Privacy-Preserving Machine Learning. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 19–38. [CrossRef]
23. Mohassel, P.; Rindal, P. ABY3: A Mixed Protocol Framework for Machine Learning. In Proceedings of the CCS '18, 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018; pp. 35–52. [CrossRef]
24. Zhao, L.; Wang, Q.; Zou, Q.; Zhang, Y.; Chen, Y. Privacy-Preserving Collaborative Deep Learning With Unreliable Participants. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 1486–1500. [CrossRef]
25. Geyer, R.C.; Klein, T.; Nabi, M. Differentially Private Federated Learning: A Client Level Perspective. *arXiv* **2018**, arXiv:1712.07557.
26. Lin, X.; Wu, J.; Li, J.; Sang, C.; Hu, S.; Deen, M.J. Heterogeneous Differential-Private Federated Learning: Trading Privacy for Utility Truthfully. *IEEE Trans. Dependable Secur. Comput.* **2023**, *20*, 5113–5129. [CrossRef]
27. Phong, L.T.; Aono, Y.; Hayashi, T.; Wang, L.; Moriai, S. Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 1333–1345. [CrossRef]
28. Park, J.; Yu, N.Y.; Lim, H. Privacy-Preserving Federated Learning Using Homomorphic Encryption With Different Encryption Keys. In Proceedings of the 2022 13th International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Republic of Korea, 19–21 October 2022; pp. 1869–1871. [CrossRef]
29. Kim, M.; Song, Y.; Wang, S.; Xia, Y.; Jiang, X. Secure Logistic Regression Based on Homomorphic Encryption: Design and Evaluation. *JMIR Med. Inform.* **2018**, *6*, e19. [CrossRef] [PubMed]
30. Kazan, G.; Kocamış, T.U. Assessing the Impact of Blockchain Technology on Internal Controls within the COSO Framework. *J. Corp. Gov. Insur. Risk Manag.* **2023**, *10*, 86–95. [CrossRef]
31. Li, J.; Shao, Y.; Wei, K.; Ding, M.; Ma, C.; Shi, L.; Han, Z.; Poor, H.V. Blockchain Assisted Decentralized Federated Learning (BLADE-FL): Performance Analysis and Resource Allocation. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 2401–2415. [CrossRef]
32. Wu, X.; Wang, Z.; Zhao, J.; Zhang, Y.; Wu, Y. FedBC: Blockchain-based Decentralized Federated Learning. In Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 27–29 June 2020; pp. 217–221. [CrossRef]
33. Lu, Y.; Huang, X.; Zhang, K.; Maharjan, S.; Zhang, Y. Low-Latency Federated Learning and Blockchain for Edge Association in Digital Twin Empowered 6G Networks. *IEEE Trans. Ind. Inform.* **2021**, *17*, 5098–5107. [CrossRef]
34. Rehman, M.H.; Salah, K.; Damiani, E.; Svetinovic, D. Towards Blockchain-Based Reputation-Aware Federated Learning. In Proceedings of the IEEE INFOCOM 2020—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada, 6–9 July 2020; pp. 183–188. [CrossRef]
35. Ramachandraiah, K.R.D.; Bommagani, N.J.; Jayapal, P.K. Enhancing healthcare data security in IoT environments using blockchain and DCGRU with Twofish encryption. *Inf. Dyn. Appl.* **2023**, *2*, 173–185. [CrossRef]
36. Wang, J.; Lin, X.; Wu, Y.; Wu, J. Blockchain-Enabled Lightweight Fine-Grained Searchable Knowledge Sharing for Intelligent IoT. *IEEE Internet Things J.* **2023**, *10*, 21566–21579. [CrossRef]
37. Lu, Y.; Huang, X.; Dai, Y.; Maharjan, S.; Zhang, Y. Blockchain and Federated Learning for Privacy-Preserved Data Sharing in Industrial IoT. *IEEE Trans. Ind. Inform.* **2020**, *16*, 4177–4186. [CrossRef]
38. 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]
39. Xu, L.; Zhang, H.; Du, X.; Wang, C. Research on Mandatory Access Control Model for Application System. In Proceedings of the 2009 International Conference on Networks Security, Wireless Communications and Trusted Computing, Wuhan, China, 25–26 April 2009; Volume 2, pp. 159–163. [CrossRef]
40. Jiang, Y.; Lin, C.; Yin, H.; Tan, Z. Security analysis of mandatory access control model. In Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583), The Hague, The Netherlands, 10–13 October 2004; Volume 6, pp. 5013–5018. [CrossRef]
41. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B.A.y. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the PMLR, 20th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017; Volume 54, pp. 1273–1282.

42. Li, T.; Hu, S.; Beirami, A.; Smith, V. Ditto: Fair and Robust Federated Learning Through Personalization. In Proceedings of the PMLR, 38th International Conference on Machine Learning, Virtual, 18–24 July 2021; Volume 139, pp. 6357–6368.
43. Elgamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory* **1985**, *31*, 469–472. [CrossRef]
44. Li, Q.; Diao, Y.; Chen, Q.; He, B. Federated Learning on Non-IID Data Silos: An Experimental Study. In Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE), Kuala Lumpur, Malaysia, 9–12 May 2022; pp. 965–978. [CrossRef]
45. Yurochkin, M.; Agarwal, M.; Ghosh, S.; Greenewald, K.; Hoang, N.; Khazaeni, Y. Bayesian Nonparametric Federated Learning of Neural Networks. In Proceedings of the PMLR, 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Volume 97, pp. 7252–7261.