Privacy-preserving Federated Learning with Lightweight and Heterogeneity in IoT

Yange Chen, Baocang Wang*, Rongxing Lu, *Fellow, IEEE,* Xu An Wang

***Abstract*—Federated learning (FL), as an emerging distributed learning framework, can combine training from different users without collecting users' original data, protecting privacy to a certain extent. However, there are no efficient privacy protection technologies applicable to IoT. One challenge in IoT is to reduce the client-server communication cost and solve communication failure questions. Another challenge is how to utilize high-quality data to guarantee training performance. To solve these challenges, we present a privacy-preserving and optimal fraction FL framework based on elliptic curve cryptosystem (ECC) and k-nearest neighbor (KNN) method in an ad-hoc network. Firstly, we propose an improved multiple key EC-ElGamal cryptosystem (MEEC), which can reduce computation overhead and improve the encryption efficiency owing to the lightweight EC-ElGamal cryptosystem with shorter keys and ciphertext. Secondly, we propose the first ad-hoc FL framework with an ad-hoc quit and join algorithm, solving the communication failure questions, guaranteeing the optimal power computation. Thirdly, we raise a Euclidean fraction scheme based on an improved KNN method, which can quickly obtain the optimal training data from the heterogeneity data, avoiding low-quality data or malicious data to join the training. Finally, security analysis and performance evaluation have been performed. Compared with the existing solutions, our scheme is secure, practicable, efficient with low communication and computational costs in IoT.**

***Index Terms*—Federated learning, k-nearest neighbor, ad-hoc network, EC-ElGamal, Euclidean fraction.**

I. Introduction

As one of the most promising technologies, federated learning (FL) enables multiple participants to conduct joint model training without sharing local data on the premise of protecting data privacy and is expected to exert great potential in the Internet of Things (IoT). To improve the intellectualization of IoT, edge computing and FL can be applied collaboratively. However, edge computing has limited computing power and communication overhead, and FL requires massive data and repeated interaction to train model, which causes the bottleneck for FL applications in edge computing of IoT.

Although FL protects privacy to a certain extent, there are no efficient privacy protection technologies applicable to IoT. As the mainstream privacy-preserving technologies, homomorphic encryption (HE) provides the operations in the ciphertext without interactions and loss of accuracy. In the current studies, homomorphic encryption schemes in FL mainly includes Paillier [1], ElGamal [2], improved BGV [3] and other homomorphic encryption algorithms [4], but these cryptosystems are not suitable for IoT devices due to their limited power resources and processing units. To avoid low efficiency and adapt the application of IoT devices, elliptic curve cryptosystem (ECC) has been improved and leveraged in our scheme because of its higher level security with shorter key size and better performance to realize the lightweight encryption, reduce resource consumption, and improve efficiency.

Owing to the features of the FL framework, FL exists high communication costs and communication failure questions. In federated learning networks, there are usually a large number of edge devices (such as intelligent robots, intelligent meters) communicating with parameter servers. If all edge devices participate in the entire training process, it will cause huge and expensive network communication overhead, and not all clients play a significant role in every round of training. In addition, network bandwidth limitations and the number of working nodes can exacerbate communication failures in federated learning, causing client device drops or exits. Among them, elliptic curves can offer an advantage to bandwidth and ad-hoc can solve the communication failures respectively.

Another bottleneck is how to utilize high-quality data. In an early study, FL treats all clients with fair data without considering the training accuracy [5], [6]. However, in real scenarios, some devices are offline or drop out, have low-quality data, unbalanced data, or non-independent and identically distributed (non-IID) data, which can cause the incorrect training result or generate a useless training model. To solve the question, Zhao *et al*. [7] proposed the first privacy-preserving deep learning framework considering the unreliable participants, which utilizes an exponential mechanism of d-ifferential privacy to learn an accurate model. However, the adversary can still recover the sensitive data and each user's data quality information is accessible to the server. Xu *et al*. [1] presented a privacy-preserving federated deep learning framework (PPFDL) to reduce the impact of irregular users and protect the privacy of all user-related information, and each user's data quality information. However, the scheme increases an additional server and a trusted third party, which leads to accessional resources and insecurity. In addition, Paillier additional homomorphism has high computational and communication costs [8].

In view of the above bottlenecks, in this paper, we present a privacy-preserving federated learning framework based on ECC and k-nearest neighbor (KNN) method in the ad-hoc

Y. Chen and B. Wang* is with the State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, 710071, China; Cryptographic Research Center, Xidian University, Xi'an, 710071, China and School of Information Engineering, Xuchang University, Xuchang, 461000, China (e-mail: ygchen428@163.com; bcwang79@aliyun.com)

R. Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada. (e-mail: rlu1@unb.ca).

X. Wang is with the Engineering University of PAP, Xi'an, 710071, China. (e-mail: wangxazjd@163.com).

network (PFLEKA) while maintaining a high-quality data utility. Our main contributions are summarized as follows.

- We present an optimal fraction FL framework with improved multiple key EC-ElGamal cryptosystem (MEEC) and ad-hoc network method suitable to IoT devices. The framework protects the privacy of all participants, reduces computational and communication costs with lightweight EC-ElGamal, and improves encryption efficiency.
- We propose ad-hoc quit and join algorithms that allow participants to quit or join the training process in the ad-hoc network, the first ad-hoc FL framework. The algorithm solves the communication failure question and guarantees the optimal power and parameter computation.
- We design a Euclidean fraction scheme based on an improved KNN method, which classifies the high-quality data to calculate the federated average. The scheme can quickly find the optimal threshold data to train the model and protect devices' data quality information.
- We conduct the security analysis and evaluate the performance of the FL scheme for convolutional neural network (CNN). The results demonstrate that our scheme is secure, practicable, efficient with low communication and computation costs.

Organization. The remaining part of this paper is organized as follows. In Section II, we discuss the related work in more detail. In Section III, some preliminaries have been presented. Section IV establishes the system and threat models. In Section V, we describe our PFLEKA in detail. In Section VI, we analyze the security of our scheme. The performance evaluation and experimental results are shown in Section VII. Finally, we draw the conclusions in Section VIII.

II. Related work

In this section, we investigate the existing works on privacy-preserving FL from the following aspects. Privacy-preserving technologies in federated learning mainly include differential privacy [9], [10], [11], [12], [13], secure multiparty computation (SMC)[11], [14], homomorphic encryption (HE) [3], [15]. In the following, we first survey the three privacy technologies in FL.

Differential privacy, as a mainstream data perturbation mechanism for privacy-preserving FL, masks the original data by adding noise. As the early FL framework, Shokri *et al.* [5] presented a distributed deep learning scheme that participants train the dataset at home and only upload the gradients to the parameter server with differential privacy. Nevertheless, the scheme still can leak the privacy [6] owing to the low privacy-preserving level. Phan *et al.* [16] proposed a privacy-preserving mechanism that can dynamically add noise based on the contribution of features without depending on the number of training steps. Gong *et al.* [17] proposed a privacy-enhanced multi-party deep learning framework by dynamically allocating privacy budgets at different stages of training. Abadi *et al.* [18] developed a framework of differential privacy by adding Laplacian noise to gradients. However, Xiang *et al.* [19] pointed out that adding noise to the gradient ensures privacy, it greatly reduces the accuracy of the model. In summary, differential privacy only provides approximately the results, reducing the data accuracy and still can leak some information in specific scenarios.

SMC performs interactive calculations to obtain convention function in the case of no trusted third party and without knowing other users' information. Bansal *et al.* [20] presented a two-party protocol using secret sharing and secure scalars. Nevertheless, the scheme lays off when the number of participants reaches a certain amount. Xu *et al.* [14] proposed an efficient approach (HybridAlpha) to protect FL privacy based on SMC protocol of function encryption. Truex *et al.* [11] developed a hybrid approach to protect FL privacy with differential privacy, SMC, and threshold HE. Bonawitz *et al.* [21] proposed a secure data aggregation for machine learning utilizing secret sharing, however, the communication overhead is too high. According to SMC's features, SMC is also not suitable for tens of thousands of clients in FL owning to more interactions and large amounts of computations.

HE allows arithmetic operations to be performed under ciphertext without loss of accuracy and interactions. Phong *et al.* [6] proposed a privacy-preserving deep learning scheme based on additively HE (Paillier and Learn- ing with Errors (LWE) cryptosystems). However, using the same private key for each participant can lead to malicious participants easily accessing other participants' private data. To overcome this shortcoming, Zhang *et al.* [22] designed a multi-key privacy-preserving deep learning scheme based on proxy re-encryption with ElGamal cryptosystem. Although DeepPAR protects the privacy of each participant with the different private keys, the scheme is not resistant to collusion attacks once the proxy server colludes with the participants to obtain the private key. Ma *et al.* [2] designed a privacy-preserving multi-party deep learning using ElGamal HE, which can protect the participants' private information. However, if the server and the participant conspire to obtain the homomorphic key, user data is no longer secure, in addition, the participants participate in joint decryption, causing the user overhead too high. Hao *et al.* [3] proposed privacy-enhanced federated learning (PEFL) to achieve efficiency with improved Brakerski-Gentry-Vaikuntanathan (BGV) HE scheme [23]. However, fully HE increases computational complexity and ciphertext length [24]. Li *et al.* [4] proposed a non-interactive privacy-preserving multi-party machine learning (NPMML) with Paillier and Rivest-Shamir-Adleman (RSA) [25] cryptosystems. Chen *et al.* [8] presented a privacy-preserving image multi-classification deep learning model based on Paillier cryptosystem. Subsequently, Chen *et al.* [26] presented a privacy-preserving deep learning model with homomorphic re-encryption based on the Bresson-Catalano-Pointcheval (BCP) [27] cryptosystem. However, their communication costs still are high.

In addition, we investigate the communication-efficient and heterogeneity problems in FL. Focusing on communication-efficient in FL, Chen *et al.* [28] proposed an asynchronous learning strategy on the clients and a temporally weighted aggregation of the local models on the server to reduce the client-server communication. Mills *et al.* [29] proposed communication-efficient FedAvg (CE-FedAvg) that can adapt

FedAvg with a distributed Adam optimization, and greatly reduce the number of convergence rounds. However, these schemes do not consider privacy concerns. Focusing on heterogeneity problems in FL, Zhao *et al.* [7] and Xu *et al.* [1] proposed privacy-preserving federated deep learning frameworks with unreliable participants in succession. However, there are some problems mentioned above. Fairness mechanisms have been proposed to solve the heterogeneity problems. Cotter *et al.* [30] introduced the proxy-Lagrangian formulation to improve fairness metrics under a rate constraint. However, setting fairness constraints can result in many users sacrificing performance to achieve fairness goals. Mohri *et al.* [31] presented an agnostic federated learning framework with data-dependent Rademacher complexity guarantees to yield a notion of fairness. However, additional knowledge to determine clients' similarities may be impractical for some IoT applications and also can cause private leakage. Pang *et al.* [32] put forward a self-organized federated learning framework for IoT with a reinforcement learning (RL) based intelligent central server, which generates a collaboration plan with a high-performance increment for clients. However, this method increases the communication cost in the collaboration plan process. In addition, when the server obtains the local parameter from all users, it triggers the aggregation process, which may cause a long waiting time. To solve the key questions of the prior works, we propose a privacy-preserving federated learning framework to improve efficiency and find the optimal score data to train the model.

III. Preliminaries

A. Secure Multiparty Computation (SMC)

SMC means that in the absence of a trusted third party, multiple parties collaboratively calculate an agreed function satisfying: $f(x_1, x_2, \ldots, x_N) = (y_1, y_2, \ldots, y_N)$, where $x_1, x_2, \ldots, x_N$ are the input, $y_1, y_2, \ldots, y_N$ are the corresponding output, and f is conventional calculation function. SMC ensures that each party only obtains its own calculation results, and it is impossible to infer any other party from the interactive data in the calculation process.

In addition, addition secret sharing has been proposed by [33], [34]. In this scheme, a sharing algorithm and a reconstruction algorithm are consisted over $\mathbb{Z}_{2^{32}}$. A secret value s is split to t shares $E_1, E_2, \ldots, E_t \in \mathbb{Z}_{2^{32}}$ satisfying

$$E_1 + E_2 + \ldots + E_t = s \bmod 2^{32}, \tag{1}$$

where $t-1$ elements $E_1, E_2, \ldots, E_{t-1}$ are uniformly distributed. This requires t participants jointly calculating their shares, and less than or equal to $t-1$ participants will not get their share.

B. Elliptic Curve

An elliptic curve E over a prime field $\mathbb{F}_p$ with the prime p ($p > 3$) is the point set of the curve following the Weierstrass equation [35]:

$$E : y^2 \bmod p = x^3 + \mathfrak{a}x + \mathfrak{b} \bmod p, \tag{2}$$

where $x, y, \mathfrak{a}, \mathfrak{b} \in \mathbb{F}_p$ and $4\mathfrak{a}^3 + 27\mathfrak{b}^2 \neq 0(\text{mod}p)$, with a point of infinity.

C. EC-ElGamal

EC-ElGamal [36], [37] is an additive homomorphic encryption scheme based on ElGamal elliptic curve, which requires a shorter key and can greatly reduce computation overhead and improve the encryption efficiency. Let $E(\mathbb{F}_p)$ be an elliptic curve over the finite field $\mathbb{F}_p$. The detailed description of the scheme is as follows:

- **KeyGen**: Given a base point $Q \in E(\mathbb{F}_p)$, choose a random integer d as the private key, and compute the public key point $P = dQ$.
- **Encryption**: First, embed the plaintext m in point M by message encoding. Then, choose the random integer r and compute the ciphertext point:

$$A_1 = M + \mathsf{r}P, A_2 = \mathsf{r}Q. \tag{3}$$

- **Decryption**: After receiving the ciphertext (A_1, A_2), decryption algorithm is computed with the private key d:

$$M = A_1 - dA_2. \tag{4}$$

  Then, M is converted to the plaintext m according to the encoding rules.
- **Additive homomorphism**: The plaintext point M_i is on the same elliptic curve $E(\mathbb{F}_p)$ with base point Q and the random integer r_i, its ciphertext point $C_i = (M_i + \mathsf{r}_i P, \mathsf{r}_i Q)$. The additive homomorphism is followed:

$$C_1 + C_2 = (M_1 + M_2 + (\mathsf{r}_1 + \mathsf{r}_2)P, (\mathsf{r}_1 + \mathsf{r}_2)Q). \tag{5}$$

D. Multiple key EC-ElGamal

At present, in many scenarios, users under different keys collaborate to obtain the best learning effect. To achieve multi-key lightweight federated learning, we propose a multi-key ECC threshold homomorphic encryption scheme. The scheme is as follows:

- **Encryption**: N participants have different private keys $\{k_1, k_2, \ldots, k_N\}$ respectively and they negotiate the same random number r utilizing SMC. In addition, they separately send the different ciphertext to the server.

$$\begin{cases} C_1 = (M_1 + rk_1Q, rQ) \\ C_2 = (M_2 + rk_2Q, rQ) \\ \cdots \\ C_N = (M_N + rk_NQ, rQ) \end{cases} \tag{6}$$

- **Secure Ciphertext Average**: The server chooses the optimal t ($2 \leqslant t \leqslant N$) ciphertext from N participants and calculates the average of t homomorphic ciphertext:

$$\begin{aligned} C =& \frac{1}{t}\sum_{i=1,j\in N}^{t}(C_j)_i \\ =& \left(\frac{1}{t}\left(\sum_{i=1,j\in N}^{t}(M_j)_i + \sum_{i=1,j\in N}^{t} r(k_j)_i Q\right), rQ\right). \end{aligned} \tag{7}$$

  Then, the server sends the average ciphertext C to t participants. Simultaneously, t participants negotiate and compute $K_i = \sum_{i=1,i\neq j}^{t}(k_j)_i$ with SMC.

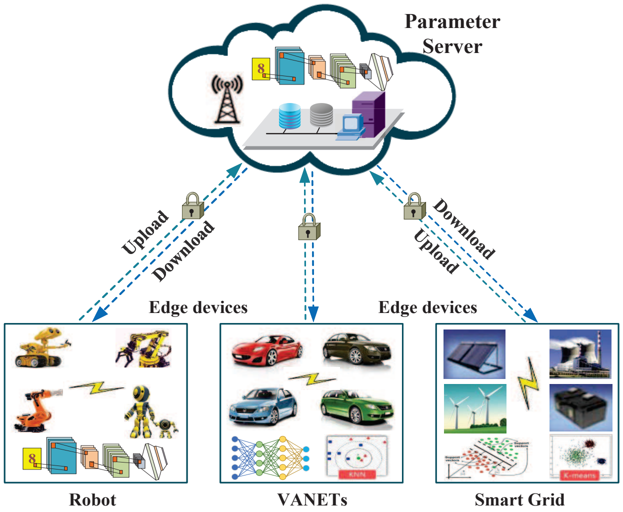


Fig. 1. System Architecture

- **Decryption**: After receiving C, the participant decrypts it with K_i and the private key $(k_j)_i$ to obtain the plaintext average M_{avg}:

$$M_{avg} = \frac{1}{t}(\sum_{i=1}^{t}(M_j)_i + \sum_{i=1}^{t} r(k_j)_i Q) - \frac{1}{t} r(K_i + (k_j)_j)Q. \quad (8)$$

IV. System model and Threat model

FL can be used for cloud computing as well as edge computing, such as the edge devices or edge intelligent network (robot, vehicular ad hoc networks (VANETs), smart grid, etc.) can be utilized FL to train and prediction data collaboratively. In this section, we give an overview of PFLEKA with ad hoc network and KNN scoring mechanism as shown in Fig. 1.

A. System model

In our system model, the boundary of edge computing has a variety of edge device domains, such as robot domain, VANETs domain, smart grid domain. In every domain, every edge device has machine learning algorithms as required, such as convolutional neural network (CNN), artificial neural network (ANN), KNN, support vector machine (SVM), k-means clustering. Devices encrypt the training parameters and upload or download the ciphertext parameters to update training data. Parameter server in the cloud or edge computing can aggregate the parameter from edge domains. In the edge network, ad hoc or other wireless network has been leveraged between edge devices and between parameters server and edge devices. We utilize the lightweight cryptosystem ECC, whose public key can be shared with all participants and the private key can be kept himself.

1) Parameter server: The parameter server, as an aggregator, can aggregate the training parameter from edge devices and calculate the federated average with the training parameter. On the server, a variety of training models can exist and multiple iterations and cooperation need to be carried out with edge devices. In addition, the server is semi-honest and cannot collude with edge devices.

2) Edge devices: an edge device is an entity that can train local data storing in the device or collecting from other devices or sensors with the needs of machine learning algorithms. First, the edge device generates the public and private key pairs. Then, it encrypts the local training weights with the private key and uploads the ciphertext weight to the parameter server. Edge devices own the private key utilized for data encryption and decryption based on ECC, and they cannot collude with the parameter server.

B. Threat model

In our PFLEKA, the parameter server is an honest-but-curious entity that can honestly follow the protocols but try to acquire the privacy information from the system model. In addition, the device generates and keeps the privacy key himself, so it prevents the leakage of the private key.

1) An internal adversary compromised by the parameter server attempts to obtain sensitive information from the training model or aggregation weights. But these information are ciphertext and the adversary cannot acquire any information.

2) An internal adversary compromised by an edge device attempts to obtain sensitive information from the other edge devices. However, the device can only receive ciphertext data and cannot obtain private information.

3) An external adversary attempts to eavesdrop on messages from internal participants or lines. Although the adversary can obtain the messages ciphertext, it cannot decrypt the messages without the private key.

V. Our proposed scheme

In this section, we give a detailed description of PFLEKA. First, a device (robot, intelligent vehicle, intelligent meter, etc.) of edge domains (robots, VANETs, Smart Grid, etc.) collects and preprocesses data from sensors or other IoT devices. Then, the device trains these data utilizing the required machine learning algorithm. After finishing the model training at home, the device encrypts model parameters using the EC-ElGamal cryptosystem, and then sends the ciphertext model parameters to the parameter server. The parameter server aggregates all ciphertext parameters (weights) from n devices with the same domain, then it calculates the average of t optimal ciphertext parameters with the federated average in FL. After obtaining the ciphertext average weights W_{avg}, the server issues these weights W_{avg} to threshold range of edge devices in the same domain. The server and edge devices train the model parameter circularly until a well-trained model is generated. Finally, the server can provide the prediction service for other users or devices, and the users or devices can predict the data it needs. In the following, we present a prime number search algorithm, an ad-hoc quit and join algorithm, a Euclidean fraction algorithm, and the detailed solution description.

A. Secure comparison protocol (SCP)

The secure comparison protocol (SCP) compares the plaintext size (M_A, M_B) of two ciphertexts C_A, C_B in server, where $C_A = M_A + rk_A Q$ and $C_B = M_B + rk_B Q$. To compare the size, two participants of two ciphtertexts participate in the comparison process. The process is followed step, and the flowchart is in Fig. 2:

Step 1: Participants A and B compute rQk_A and rQk_B separately. Participant A generates the random number r_A and calculates and sends $rQk_A + r_A$ to participant B.

Step 2: Participant B computes and obtains $C_R = rQ(k_B - k_A) - r_A$, and then sends to the parameter server.

Step 3: The server computes and sends $CC = C_A - C_B + C_R$ to participant A.

Step 4: Participant A adds the random number r_A to CC to obtain $M_A - M_B$. Judging $M_A - M_B$ size, Participant A sends the size result to the server.

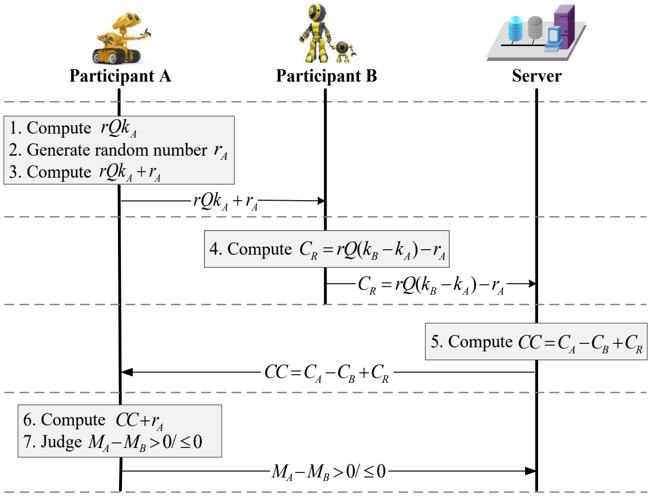

Participant A
Participant B
Server
1. Compute rQk_A
2. Generate random number r_A
3. Compute $rQk_A + r_A$
$rQk_A + r_A$
4. Compute $C_R = rQ(k_B - k_A) - r_A$
$C_R = rQ(k_B - k_A) - r_A$
5. Compute $CC = C_A - C_B + C_R$
$CC = C_A - C_B + C_R$
6. Compute $CC + r_A$
7. Judge $M_A - M_B > 0 / \le 0$
$M_A - M_B > 0 / \le 0$

Fig. 2. Secure comparison protocol

B. Secure multiplication protocol (SMP)

Given two ciphertexts $E_{pk_i}(m_1) = (A_{1i}, A_{2i}) = (m_1 + rk_i Q, rQ)$ and $E_{pk_i}(m_2) = (A_{1i}{}', A_{2i}{}') = (m_2 + rk_i Q, rQ)$, calculate $E_{pk_i}(m_1 m_2)$ by the secure multiplication protocol (SMP). The detailed process is followed:

Step 1: The participant i encrypts m_1 to obtain the ciphertext $E_{pk_i}(m_1)$ with the public key $pk_i = P_i = k_i Q$, and then computes and sends $D = (m_1 - 1)rP_i$ and $F = rP_i$ to the server;

Step 2: The server encrypts m_2 to obtain the ciphertext $E_{pk_i}(m_2)$ with the public key $pk_i = k_i Q$.

Step 3: The server calculates $E_{pk_i}(m_1 m_2)$ as follows:

$$\begin{aligned} &E_{pk_i}(m_1 m_2) \\ =&(A_{1i} A_{1i}{}' - F^2 - D - m_2 F, A_{2i}) \\ =&(m_1 m_2 + rk_i Q, rQ). \end{aligned} \tag{9}$$

C. Prime number search algorithm (PNSA)

ECC key size is the main parameter to optimize performance and reduce power consumption, which can achieve lightweight encryption and low communication delay. In IoT, due to the limited battery life of the sensors or devices, sensors or devices need low power consumption to operate, and the optimal prime p can reduce the low power consumption to a great extent. Therefore, choosing the appropriate prime p with the acceptable security level is crucial [21]. To find the optimal prime p, we propose the following prime number search algorithm.

Let local epoch power be P_e, f be the number of local epoch, the communication power be P_c. A round of the total power P_t in FL is the summation of local processing power and the communication power as follows:

$$P_t = fP_e + P_c. \tag{10}$$

The total power P_{total} satisfies the following formula.

$$P_{total} = mP_t < P_{threshold}, \tag{11}$$

where m is the optimal training round when the weights are not updated again, or is the threshold value of the round, and $P_{threshold}$ is the threshold value of the power. When reaching the threshold value $P_{threshold}$, the device will be shut down owing to the battery running out. The algorithm is followed in Algorithm 1. In the algorithm, $O(p)$ is the order of the prime p satisfying $O(p) \geqslant O_{\min}$, where $O_{\min}$ is the order of the minimum security level.

Algorithm 1: Prime number search algorithm

```
Input: $p \leftarrow 1, P_{avg} \leftarrow \infty, O(p) \leftarrow 0$
Output: prime $p$

Compute $P_{avg} = \frac{1}{t}(\sum_{i=1}^{t} P_{total}(i))$;
    if $P_{avg} > P_U$ (predefined power budget)
    then choose the next prime $p$;
        compute the average power $P_{avg}$;
    else compute $O(p)$ with Schoof algorithm [38];
        if $O(p) < O_{\min}$
        then choose the next prime $p$;
        else obtain the value of prime number $p$.
        end if
    end if
```

D. Euclidean fraction algorithm (EFA)

To obtain the optimal federated training model by calculating the federated average of t optimal weights, we present a Euclidean fraction algorithm with an improved KNN method, namely the KNN scoring mechanism. The steps are as follows:

Step 1: The server receives $E_{pk_i}(W_i) = (E_{pk_i}(w_{i1}), E_{pk_i}(w_{i2}), \ldots, E_{pk_i}(w_{ij}), \ldots, E_{pk_i}(w_{in}))$ from participant i with the number of weights n and encrypts $(w_1{}', w_2{}', \ldots, w_n{}')$ to obtain the ciphertext $E_{pk_i}(W_i{}') = (E_{pk_i}(w_1{}'), E_{pk_i}(w_2{}'), \ldots, E_{pk_i}(w_n{}'))$, where $(w_{i1}, w_{i2}, \ldots, w_{in})$ is the training dataset, $(w_1{}', w_2{}', \ldots, w_n{}')$ is the test dataset, $i \in \{1, N\}, j \in \{1, n\}$.

Step 2: The server computes the distance between the training dataset and the test dataset using Euclidean distance, which can be written as

$$\begin{aligned} &d(W_i, W_j{}') \\ =&\sqrt{(w_{i1} - w_1{}')^2 + (w_{i2} - w_2')^2 + \ldots + (w_{in} - w_n{}')^2}. \end{aligned} \tag{12}$$

Since we obtain the ciphertext training dataset and test dataset, we first calculate $E_{pk_i}((w_{ij}-w_j{}')^2)$ with the additive homomorphism and SMP protocol as follows:

$$\begin{aligned}&E_{pk_i}((w_{ij}-w_j{}')^2)\\=&E_{pk_i}((w_{ij})^2)-E_{pk_i}(2w_{ij}w_j{}')+E_{pk_i}((w_j{}')^2).\end{aligned} \tag{13}$$

Then we calculate the square of the distance with the additive homomorphism, namely:

$$\begin{aligned}&E_{pk_i}(d^2(W_{ij},W_j{}'))\\=&E_{pk_i}((w_{i1}-w_1{}')^2)+E_{pk_i}((w_{i2}-w_2{}')^2)\\&+\ldots+E_{pk_i}((w_{in}-w_n{}')^2).\end{aligned} \tag{14}$$

Step 3: The server sorts the square of the distances $E_{pk_1}(d^2(W_{1j},W_j{}')),\ldots,E_{pk_n}(d^2(W_{nj},W_j{}'))$ under ciphertext with the SCP from the biggest value to the smallest value, and then takes the biggest k points written as $(E_{pk_{a1}}(d_1),E_{pk_{a2}}(d_2),\ldots,E_{pk_{ak}}(d_k))$ corresponding to $(E_{pk_{a1}}(W_{d_1}),E_{pk_{a2}}(W_{d_2}),\ldots,E_{pk_{ak}}(W_{d_k}))$, where $a1,a2,\ldots,ak$ is the subscript of the public key after sorting and $d_1,d_2,\ldots,d_k$ is the distance of k optimal fraction algorithm from $d^2(W_{ij},W_j{}')$.

Step 4: Finding the k original ciphertexts $E_{pk_i}(W_i)$ of k optimal distances from N participants, the server computes the federated average $E(W_{avg})$ using the k optimal weights with multiple key.

$$\begin{aligned}&E(W_{avg})\\=&\frac{1}{k}(E_{pk_{a1}}(W_{d_1})+E_{pk_{a2}}(W_{d_2})+\ldots+E_{pk_{ak}}(W_{d_k}))\end{aligned} \tag{15}$$

E. Construction of PFLEKA

In this subsection, we give a detailed description of PFLEKA in Fig. 3. According to the PFLEKA workflow, the scheme includes the following phases: the initialization and encryption phase, the ad-hoc network establishment phase, the federated training phase, the decryption and prediction phase.

Initialization and encryption phase: Participant i (robot, vehicle, smart meter etc.) collects and preprocesses the data to obtain dataset $(D_1,D_2,\ldots,D_u)$ from various of sensors, where u is the number of the dataset. Then, the participant i trains these dataset $(D_1,D_2,\ldots,D_u)$ and obtains the weights. Before uploading these weights, the edge device operates the following operation. First, take the edge device as participant i, participant i generates the private key k_i using the key generation and computes and sends the public key point $pk_i=P_i=k_iQ$ to the parameter server. Then, N participants negotiate and obtain the same random number r utilizing SMC.

After obtaining the initial weight W_0, participant i trains the dataset $(D_1,D_2,\ldots,D_u)$ with machine learning algorithm such as CNN. When obtaining the training weight $W_i=(w_{i1},w_{i2},\ldots,w_{in})$, participant i encrypts the weights W_i with the private key k_i and the random number r utilizing multiple key EC-ElGamal cryptosystem to obtain $E_{pk_i}(W_i)=(E_{pk_i}(w_{i1}),E_{pk_i}(w_{i2}),\ldots,E_{pk_i}(w_{in}))$. Finally, all participants obtain the ciphertext weights $E_{pk_i}(W_i)$ and send them to the aggregation server.

Ad-hoc network establishment phase: In our scheme, to make routing protocol more bandwidth efficient, we leverage Dynamic Source Routing (DSR) [39] with the ECC in IoT to establish the ad-hoc network. DSR [40] is an ad-hoc on-demand routing protocol that establishes the routed protocol containing the address of the packet traversing nodes by flooding RouteRequest packets in a source node and sending a RouteReply packet to the source node from the destination node. DSR avoids transmitting the same RouteRequest repeatedly by a source node or an intermediate node, and the final path includes the address of each device.

In the ad-hoc network establishment phase, we propose ad-hoc quit and join algorithms suitable for the establishment process. When the ith node joins and quits from the ad-hoc network, the network updates the route with DSR protocol. If the ith node quits, the parameter server receives the quit signal of the ith node, removes its related weights $E(W_i)$. If the quit node exists in threshold t nodes ($i\in t,t\geqslant 3$), then the server computes $t-1$ ciphertext weight average $E(W_{avg})$ and the average power P_{avg} with the proposed multiple key EC-ElGamal cryptosystem in formula (16). Otherwise, the server still computes the original t ciphertext.

$$\begin{aligned}E(W_{avg})=&\frac{1}{t-1}(E(W_1)+\cdots+E(W_j)+\cdots+E(W_t))(j\neq i)\\=&\frac{1}{t-1}(\sum_{j=1,j\neq i}^{t}W_j+\sum_{j=1,j\neq i}^{t}r_iP,\sum_{j=1,j\neq i}^{t}r_iQ)\\=&\frac{1}{t-1}(\sum_{j=1,j\neq i}^{t}E(W_j))\end{aligned} \tag{16}$$

If the i_* node joins, the parameter server receives the ciphertext weights $E(W_{i_*})$ of the i_* node, compares the smallest ciphertext weight $E(W_t)$ sorted by SCP with $E(W_{i_*})$. When judging $W_{i_*}>W_t$ with the SCP, the server replaces $E(W_t)$ with $E(W_{i_*})$ and computes the average ciphertext $E(W_{avg})$ with the proposed multiple key EC-ElGamal cryptosystem and the power average P_{avg}. The average ciphertext $E(W_{avg})$ is followed:

$$\begin{aligned}E(W_{avg})=&\frac{1}{t}(E(W_1)+E(W_2)+\cdots+E(W_{t-1})+E(W_{i_*}))\\=&\frac{1}{t}(\sum_{j=1}^{t-1}E(W_j)+E(W_{i_*})).\end{aligned} \tag{17}$$

When judging $W_{i_*}\leqslant W_t$ with the SCP, the server keeps the original training process with the optimal t ciphertext weight.

Federated training phase: The parameter server obtains N participants's $E_{pk_i}(W_i)$, judges the optimal t ciphertext with EFA, and then computes the federated average with multiple key EC-ElGamal encryption. The detailed federated learning algorithm is followed:

Finally, through the global update and local update circularly, the server and the participants obtain the well-trained model when reaching the round threshold or the optimal weight without an update again.

Decryption and predication phase: After obtaining the well-trained model $E_{pk}(W^*_{avg})$, the participant decrypts

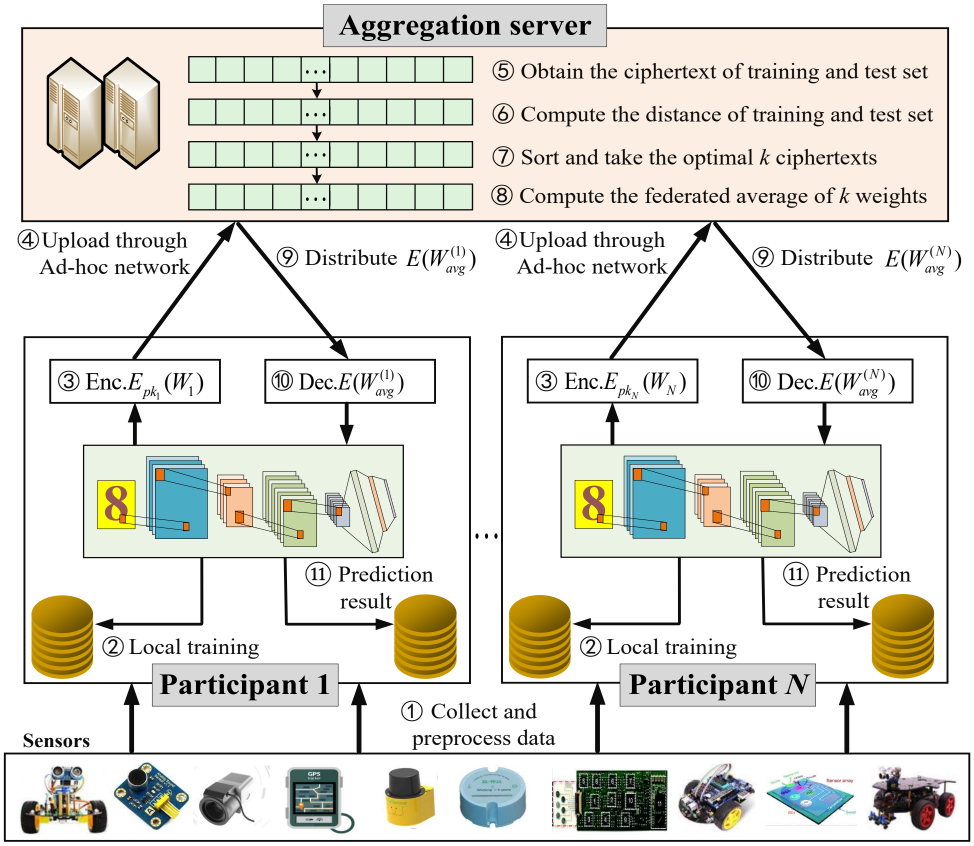

Aggregation server
⑤ Obtain the ciphertext of training and test set
⑥ Compute the distance of training and test set
⑦ Sort and take the optimal k ciphertexts
⑧ Compute the federated average of k weights
④Upload through Ad-hoc network
⑨ Distribute $E(W_{avg}^{(1)})$
④Upload through Ad-hoc network
⑨ Distribute $E(W_{avg}^{(N)})$
③ Enc.$E_{pk_1}(W_1)$
⑩ Dec.$E(W_{avg}^{(1)})$
③ Enc.$E_{pk_N}(W_N)$
⑩ Dec.$E(W_{avg}^{(N)})$
8
8
⑪ Prediction result
⑪ Prediction result
② Local training
② Local training
Participant 1
Participant N
...
① Collect and preprocess data
Sensors

Fig. 3. PFLEKA Workflow

Algorithm 2: Ad-hoc quit algorithm

```
Input: $i$th node
Output: $P_{avg}$ and $E(W_{avg})$
Participant:
    if the $i$th node quits;
    then update the route with DSR;
    end if
Server:
    Receive the quit signal of $i$th nodes from ad-hoc network;
    Delete $E(W_i)$;
    if the quit device $i \in t$;
    then Compute $t-1(t \geqslant 3)$ ciphertext weights average:
$E(W_{avg}) = \frac{1}{t-1}(\sum_{j=1, j\neq i}^{t} E(W_j))$ with multiple key
EC-ElGamal homomorphism.
    Compute $t-1$ power average:
    $P_{avg} = \frac{1}{t-1}(\sum_{j=1, j\neq i}^{t} P_{total}^{(j)})$.
    else Compute $t$ ciphertext weights average:
$E(W_{avg}) = \frac{1}{t}(\sum_{j=1}^{t} E(W_j))$ with multiple key EC-ElGamal
homomorphism;
    Compute $t$ power average: $P_{avg} = \frac{1}{t}(\sum_{j=1}^{t} P_{total}^{(j)})$.
    end if
```

$E_{pk}(W_{avg}^{*})$ to obtain W_{avg}^{*}. Then these participants can predict the result according to the well-trained model and their dataset. The other prediction users can predict the result in the parameter server with the well-trained model.

Algorithm 3: Ad-hoc join algorithm

```
Input: $i_*$th node
Output: $P_{avg}$ and $E(W)$
Participant:
    if the $i_*$th node joins;
    then update the route with DSR;
        Upload the $i_*$th ciphertext weight $E(W_{i_*})$ to the
    parameter server.
Server:
    Compare the small ciphertext weight $E(W_t)$ with the new
ciphertext weight $E(W_{i_*})$;
    if $W_{i_*} > W_t$ with the SCP;
    then replace $E(W_t)$ with $E(W_{i_*})$;
        Compute $E(W) = \frac{1}{t}(\sum_{i=1}^{t-1} E(W_i) + E(W_{i_*}))$
    with multiple key EC-ElGamal homomorphism;
        Compute $P_{avg} = \frac{1}{t}(\sum_{j=1}^{t} P_{total}^{(j)})$;
    else keep the original training process.
    end if
```

VI. Security Analysis

In this section, we perform the security definition, semantic security, and security analysis.

Definition 1: Elliptic curve discrete logarithm problem (ECDLP): Given two points $P, Q \in E(\mathbb{F}_p)$ and $d \in \mathbb{Z}_p^*$, calculating $P = dQ$ is easy. But if we know P and Q,

Algorithm 4: the optimal fraction FL framework (OFFLF)

```
GlobalUpdate:
Initialize the initial weight $W_0$;
Broadcast the initial weight $W_0$ to every participant;
for each round do
    LocalUpdate($W_i$);
end for
Get $E_{pk_i}(W_i), E_{pk_i}(W_j{}')$ according to EFA algorithm;
for every parameter do
    Compute the distance $E_{pk_i}((w_{ij} - w_j{}')^2)$ with SMP;
end for
$E_{pk_i}(d^2(w_{ij}, w_j{}')) = \sum_{j=1}^{n} E_{pk_i}((w_{ij} - w_j{}')^2)$;
Sort $E_{pk_i}(d^2(W_{ij}, W_j{}'))$ with SCP;
Take the biggest $t$ points $\{E_{pk_{ai}}(W_{d_i})\}_{i=(1,2,\ldots,t)}$;
Calculate $E(W_{avg})$ with multiple key EC-ElGamal;
Send $E(W_{avg})$ to the $t$ parameters.

LocalUpdate($W_i$):
Get $E(W_{avg})$ from the server;
Decrypt to obtain the weight $W_{avg}$;
for each epoch do
    Compute the gradient $G$;
    $W_i \leftarrow W_{i-1} - \eta G$ ($\eta$ is the learning rate);
    Encrypt to obtain $E_{pk_i}(W_i)$;
    Send $E_{pk_i}(W_i)$ to the server.
end for
```

solving d is hard according to the discrete logarithm problem for elliptic curves, in that there is no known polynomial time algorithm that can run on classical computers.

Definition 2: Elliptic Curve Decisional Diffie-Hellman Problem (ECDDH): Given Q and $aQ, bQ \in E(\mathbb{F}_p)$, it is hard to distinguish between the two distributions (aQ, bQ, abQ) and (aQ, bQ, cQ), where $a, b, c \in \mathbb{Z}_q^*$.

A. Semantic Security

We demonstrate that the proposed cryptosystems are semantically secure under the DDH assumption.

Theorem 1. EC-ElGamal is indistinguishability from chosen-plaintext attack (IND-CPA) under DDH assumption. *Proof*: In the EC-ElGamal, we denote the first part of ciphertext $c_1 = M + aQ$. A simulator S can be constructed against the ECDDH problem from an adversary $\mathcal{A}$. We design the following game between the simulator S and adversary $\mathcal{A}$. In the game, the advantage of the $\mathcal{A}$ is $\varepsilon = Pr[\mathfrak{b}' = \mathfrak{b}] - \frac{1}{2}$ and the advantage of the S is that ε' is not less than ε. Considering the following algorithm as a challenger, the challenger flips a coin $\mu \in \{0, 1\}$. If $\mu = 0$, the distribution is (aQ, bQ, abQ); If $\mu = 1$, the distribution is (aQ, bQ, cQ). The algorithm is followed:

1) Given $P = dQ$ and set $pk = \{E(\mathbb{F}_p), P, Q\}$, the S contacts the $\mathcal{A}$ to obtain two message m_0 and m_1 in $\mathbb{Z}_q^*$.
2) The S encrypts $m_\mathfrak{b}$ for $(aQ, m_\mathfrak{b} + abQ)$ by flipping the coin $\mathfrak{b} \in \{0, 1\}$ and sends the ciphertext to the $\mathcal{A}$.
3) The $\mathcal{A}$ guesses the value of $\mathfrak{b}$ as $\mathfrak{b}'$, and then sends to the S.
4) The S output $\mu' = 0$ if $\mathfrak{b} = \mathfrak{b}'$ to manifest (aQ, bQ, abQ) when $\mu = 0$ and the $\mathcal{A}$ wins the game with the advantage ε. Otherwise, the S output $\mu' = 1$.

Therefore, when $\mu = 0$ and $\mathfrak{b} = \mathfrak{b}'$, the $\mathcal{A}$ can obtain $cQ = abQ$, namely, the $\mathcal{A}$ has:

$$\Pr[\mu = 0 | \mathfrak{b} = \mathfrak{b}'] = \frac{1}{2} + \varepsilon.$$

The S satisfies:

$$\Pr[\mu' = \mu | \mu = 0] = \frac{1}{2} + \varepsilon.$$

When $\mu = 1$, the $\mathcal{A}$ cannot acquire any information about $m_\mathfrak{b}$, namely, the $\mathcal{A}$ has:

$$\Pr[\mu = 1 | \mathfrak{b} \neq \mathfrak{b}'] = \frac{1}{2}.$$

The S satisfies:

$$\Pr[\mu' = \mu | \mu = 1] = \frac{1}{2}.$$

Ultimately, the advantage of the S is as follows:

$$\begin{aligned}\varepsilon' =& \frac{1}{2}\left(\Pr[\mu' = \mu | \mu = 0] + \Pr[\mu' = \mu | \mu = 1]\right) - \frac{1}{2} \\ =& \frac{\varepsilon}{2} < \varepsilon. \end{aligned} \tag{18}$$

The result conflicts with our assumption, therefore, EC-ElGamal is secure under the DDH assumption. The reason is as follows:

In the key generation of the EC-ElGamal cryptosystem, the public key P can be calculated in ECDLP, so the process is secure. In the encryption, the first part of the ciphertext satisfies ECDDH and the second part is guaranteed by ECDLP. In the decryption, the security is ensured by ECDDH.

Theorem 2. If EC-ElGamal is semantically secure, then multiple key EC-ElGamal is semantically secure.

Proof: In multiple key EC-ElGamal, according to theorem 1, the encryption process is semantic security in formula (5) under ECDLP and ECDDH owing to the advantage of the $\mathcal{A}$ cannot reach ε. In secure ciphertext average, the ciphertext in formula (6) from participants maintains the semantic security due to participants' semantic security, and K_i negotiated with SMC is secure owing to SMC's definition to protect private information from different users. After receiving the ciphertext C, the decryption still guarantees security under ECDDH.

B. Security Analysis

Theorem 3. SCP security can be guaranteed if EC-ElGamal is semantically secure.

Proof: In SCP protocol, all data are based on EC-ElGamal encryption, discrete logarithm, and adding the random masking. Among them, EC-ElGamal encryption protects the confidentiality owing to its semantically secure, the discrete logarithm is hard, and the random masking has add the blinding factor r_A and avoids the participants obtaining any information. subsequently, we analyze the security of the SCP protocol in the ideal world and real-world game.

In the SCP, the adversary $\mathcal{A}$ interacts with participants running protocol π in the environment $\mathcal{Z}$. The view of $\mathcal{A}$ in the real world is:

$$V_{\text{Real}} = \{rQk_A, rQk_B, rQk_A + r_A, rQ(k_B - k_A) - r_A, CC\}.$$

In the ideal world, a simulator S replacing participants is built to interact with the adversary $\mathcal{A}$, and produces the same number of random numbers. The view of the S in the ideal world is:

$$V_{\text{Ideal}} = \{[r_{11}], [r_{12}], [r_{11} + r_{13}], [r_{12} - r_{11} - r_{13}], [r_{14}]\}.$$

Owing to the semantic security of EC-ElGamal, we say that the protocol π can realize an ideal functionality F if the adversary $\mathcal{A}$ and the S exist:

$$\{\text{IDEAL}^{SCP}_{F,S,\mathcal{Z}}(V_{\text{Ideal}})\} \approx \{\text{REAL}^{SCP}_{\pi,\mathcal{A},\mathcal{Z}}(V_{\text{Real}}\}.$$

Theorem 4. SMP security can be guaranteed if EC-ElGamal is semantically secure.

Proof: In SMP protocol, all data are based on EC-ElGamal encryption and discrete logarithm. In the SMP, the adversary $\mathcal{A}$ interacts with participants running protocol π in the environment $\mathcal{Z}$. The view of $\mathcal{A}$ in the real world is:

$$V_{\text{Real}}{}' = \{m_1 + rk_iQ, m_2 + rk_iQ, rQ, D, F, E_{pk_i}(m_1m_2)\}.$$

Due to the semantic security of EC-ElGamal, the S generates the same number of random ciphertexts. The view of $\mathcal{A}$ satisfies the following formula:

$$V_{\text{Ideal}}{}' = \{[r_{21}], [r_{22}], [r_{23}], [r_{24}], [r_{25}], [r_{26}]\}.$$

The adversary $\mathcal{A}$ cannot distinguish the ideal world from the real world in ideal functionality F:

$$\{\text{IDEAL}^{SMP}_{F,S,\mathcal{Z}}(V_{\text{Ideal}})\} \approx \{\text{REAL}^{SMP}_{\pi,\mathcal{A},\mathcal{Z}}(V_{\text{Real}}\}.$$

Theorem 5. EFA security can be guaranteed if EC-ElGamal is semantically secure.

Proof: In EFA, all process data are based on EC-ElGamal. In step 1, the encrypted test dataset is known by the server, so the plaintext and ciphertext are not affecting their security. In step 2, calculating $E_{pk_i}((w_{ij} - w_j{}')^2)$ and $E_{pk_i}(d^2(W_{ij}, W_j{}'))$ are based on the additive homomorphism of EC-ElGamal and SMP protocol, and SMP security is also based on EC-ElGamal, which has been proofed in theorem 4. Therefore, the process is semantic security. In step 3, the server sorting the process of the square of the distance under ciphertext utilizes the SCP protocol, which has been proofed the security in theorem 3. Hence, the security of step 3 is guaranteed. The federated average $E(W_{avg})$ is calculated under multiple key EC-ElGamal, whose security is guaranteed in theorem 2. Therefore, step 4 is semantic security.

Theorem 6. The proposed optimal fraction FL framework is secured against reconstruction attacks.

Proof: Assume that a malicious adversary such as the server or an external attacker who simulates the server function might obtain the training parameters from different edge devices. In plaintext, when data structure or figure has been known, the weights or gradients can leak the privacy information from edge devices or users [6]. In our scheme, the weights are ciphertext under EC-ElGamal and the server can only black box access. According to the semantic security of EC-ElGamal, the optimal fraction FL framework withstands reconstruction attack.

Theorem 7. The proposed scheme resists collusion attacks under certain conditions.

Proof: In our scheme, the edge device and server do not collude. Once colluded, the server can obtain the device's private key and calculate the sum of K_i and k_i to decrypt the average weight, which may leak the privacy. Therefore, we assume that the edge device and server can not collude. For the collusion question during the edge devices, it guarantees privacy even though the majority of participants (edge devices) under the condition $\{2 \leqslant i < t\}$ are corrupted in multiple key EC-ElGamal. Therefore, the proposed scheme can withstand collusion attacks under certain conditions.

VII. PERFORMANCE EVALUATIONS

In this section, we evaluate and experiment with the performance and accuracy of PFLEKA. First, we discuss the communication and computational costs. Secondly, we analyze and test the performance of the elliptic curve and cryptosystem. Finally, we assess the accuracy of our improved FL scheme with the KNN method.

A. Complexity analysis

In the subsection, to simplify the representation of the communication and computational costs, we denote an encryption/decryption as Enc/Dec, a point multiplication/division as Mul/Div.

Communication cost. In the initialization and encryption phase, N participants send nNEnc + NMul communication costs. In the ad-hoc network establishment phase, the ad-hoc quit algorithm removes a node, it only sends a quit signal and generates 1 communication cost. The ad-hoc join algorithm joins one or more nodes, and a new node generates Enc + 4Mul costs to communicate with the parameter server. In the federated training phase, the parameter server costs Enc + 5Mul to a participant. In the decryption and prediction phase, there are no cost during the server and participant. The communication cost of our scheme is summarized in Table I.

TABLE I
COMMUNICATION COST

Phase	Communication cost
Initialization and encryption	nNEnc + NMul
Ad-hoc network (quit)	1
Ad-hoc network (join)	Enc + 4Mul
Federated training	Enc + 5Mul
Decryption and prediction	–

Computational cost. In the initialization and encryption phase, participant costs Mul + Enc to compute the public key and encryption. In the ad-hoc network establishment phase, the ad-hoc network quit algorithm takes 0 costs. In the ad-hoc network join algorithm, participant costs nEnc to compute the ciphertext weights, and the parameter server takes 4Mul costs to compare the SCP. In the federated training phase, participant costs 7Mul to assist computing EFA algorithm with SMP and SCP, and the server takes nEnc + $16n$Mul + Div to compute the optimal fraction FL algorithm. In the decryption and prediction phase, participant takes Dec cost to decrypt the ciphertext result. The computational cost of our scheme is followed in Table II.

TABLE II
COMPUTATIONAL COST

Phase	Participant	Server
Initialization and encryption	Mul + Enc	–
Ad-hoc network (quit)	–	0
Ad-hoc network (join)	nEnc	4Mul
Federated training	7Mul	nEnc + 16nMul + Div
Decryption and prediction	Dec	–

B. Performance analysis

In cryptosystems, RSA and ElGamal algorithms are not suitable for environments with limited bandwidth (e.g. IoT devices, sensors, robots with limited functions) because of their high computational costs in computing exponential operations. ECC needs only additions and multiplications to calculate the encryption operations, can achieve the security requirement with the short key size, decreases computation cost greatly, and improves the encryption efficiency. Since ECC as a universal cryptosystem in IoT devices is applied in our scheme. The key lengths of common cryptosystems are compared in Table III.

TABLE III
KEY LENGTH COMPARISON

Level	Paillier	RSA	ECC	ElGamal
80	1024	1024	160	1024
112	2048	2048	224	2048
128	3072	3072	256	3072
192	7680	7680	384	7680
256	15360	15360	512	15360

To verify and simulate the performance and accuracy, PFLEKA is tested using an Intel(R) Core(TM) i7-7700HQ CPU @2.80GHz(8CPUs), 8GB RAM, and the 64-bit Windows operating system with Anaconda 3, PyCharm 2020.3.2 Professional Edition, Python 3.8.5, and PyTorch 1.7.0. We adopt the mnist dataset and ECC cryptosystem with Curve25519. More users are realized by the split dataset method.

To compare the efficiency, we first test the running time of encryption and decryption and the point multiplication based on EC-ElGamal. The running time is followed in Table IV.

TABLE IV
THE RUNNING TIME OF EC-ELGAMAL

Encryption	Decryption	Point multiplication
214.43 ms	106.71 ms	0.17ms

According to the running time of EC-ElGamal, we compare the communication and computational costs in Fig. 4. Fig. 4(a) compares the running time of the communication cost with the number of weights n variation {10, 20, . . . , 100}, where the number of participants is 10. As shown in Fig. 4(a), the time in the ad-hoc network (join) and federated training phases are almost the same. In the initialization and encryption phase, the running time of communication cost increases obviously as the number of weights grows. Fig. 4(b) compares the running time of communication cost with the number of participants increasing. As shown in Fig. 4(b), as the number of participants {10, 20, . . . , 100} grows, in the initialization and encryption phase, the running time of communication cost increases constantly, and the time retain unchanged in the ad-hoc network (join) and federated training phases. The results show that communication costs are affected in the initialization and encryption phase mainly. Fig. 4(c) and Fig. 4(d) compare the running time of computational costs with the number of weights n increasing. In Fig. 4(c), the computational running time of participants in the ad-hoc network (join) phase augments as the number of weights {10, 20, . . . , 100} increases. The computational cost in other phases is not affected by the number of weights. In Fig. 4(d), the computational running time of the server in the federated training phase increases ceaselessly as the number of weights {10, 20, . . . , 100} grows, and the running time in other phases is almost 0 or 0. The results show that the computational cost of participants is affected in the ad-hoc network (join) phase and the computational cost of the server is affected in the federated training phase.

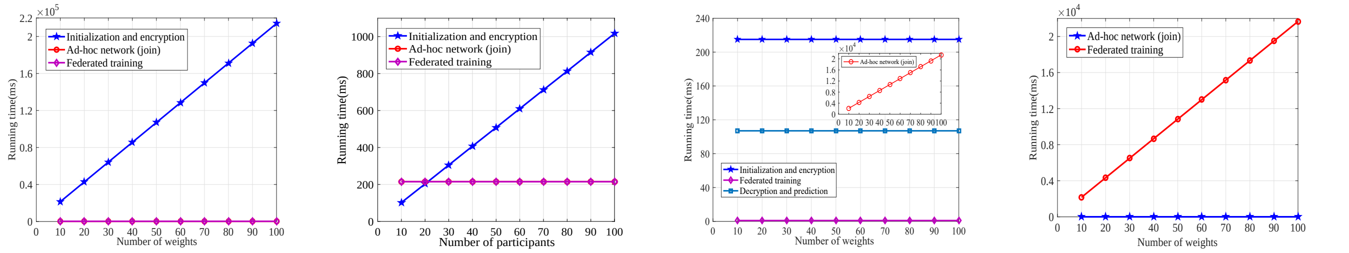
To experiment performance efficiency of our scheme, we first need to decide which ECC to use. We experiment with several elliptic curves includes P256, secp256k1, Curve25519, M383, E222, E382, and the results show that Curve25519 is a better choice. In addition, Curve25519 is recommended by the National Institute of Standards and Technology (NIST) and the Internet Engineering Task Force (IETF) suitable for higher-level security requirements. Therefore, we choose Curve25519 as the elliptic curve of the ECC cryptosystem. The performance comparison of elliptic curves is followed in Fig. 5.

Fig. 5(a) tests the key generation time of five times under several elliptic curves, and Table V computes their averages of key generation time. In Fig. 5(a), as the number of images {1, 2, . . . , 5} increases, the running time of P256, Curve25519, E222 varies slightly, while the running time of P256, secp256k1 is almost the same. The values of M383 and E382 in the top x axis see the bottom of the histogram, whose values are (102, 171). The other histograms utilize the same method to observe. In Figs. 5(b), 5(c), as the number of images {1, 2, . . . , 5} increases, the encryption and decryption time augments constantly. To reduce the encryption and decryption time, we adopt the removing zero method for weights to encrypt and decrypt. Figs. 6(a), 6(b) show that the encryption and decryption time also increases continually with the number of images growing, but the time of Figs. 6(a), 6(b) are obviously lower than Figs. 5(b), 5(c).

TABLE V
KEY GENERATION OF ELLIPTIC CURVE

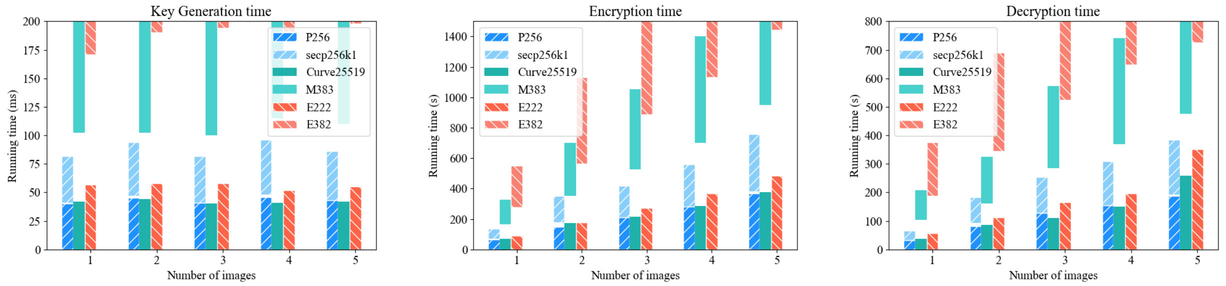
Curve type	Key generation time
P256	45 ms
secp256k1	44 ms
Curve25519	42 ms
M383	104 ms
E222	56 ms
E382	189 ms

After confirming the elliptic curve, we compare the common cryptosystems {LWE, Paillier, ElGamal} with our scheme. In Table VI, we first test the key generation time of cryptosystems, and the time of LWE is significantly longer than



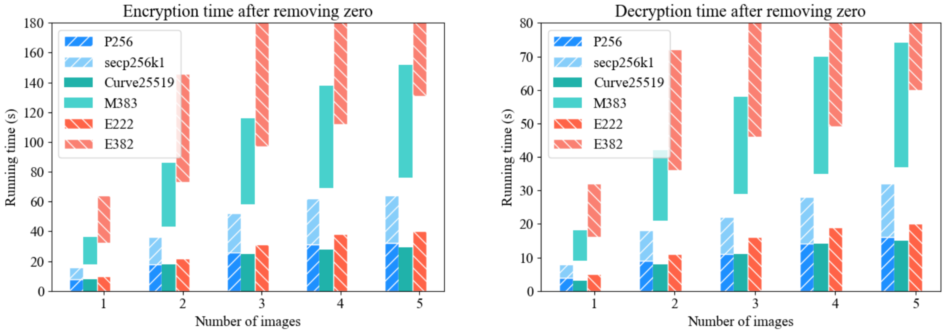
(a) Communication cost with weight variation (b) Communication cost with participant variation (c) Computational cost of participant with weight variation (d) Computational cost of the server with weight variation

Fig. 4. Communication and Computational costs



(a) Key Generation time (b) Encryption time (c) Decryption time

Fig. 5. Performance analysis of elliptic curve



(a) Encryption time (b) Decryption time

Fig. 6. Performance analysis of elliptic curve

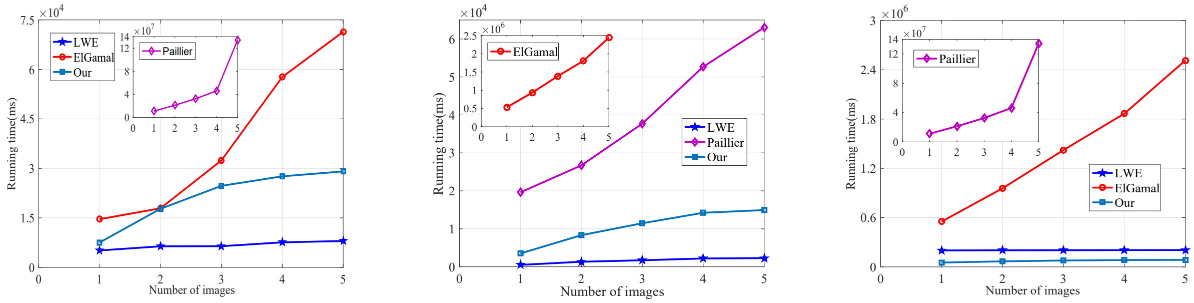
other cryptosystems. In Figs. 7(a), 7(b), 7(c), as the number of images $\{1, 2, \ldots, 5\}$ increases, the running times of Paillier and ElGamal rise distinctly, and LWE's and ours increase slightly. Although the running time of our encryption and decryption is longer than LWE, the total time is lower than LWE owing to the high key generation time of LWE, as shown in Fig. 7(c).

TABLE VI
KEY GENERATION OF CRYPTOSYSTEM

Cryptosystem	Key generation time
LWE	195 s
Paillier	2982 ms
ElGamal	2636 ms
Our	42 ms

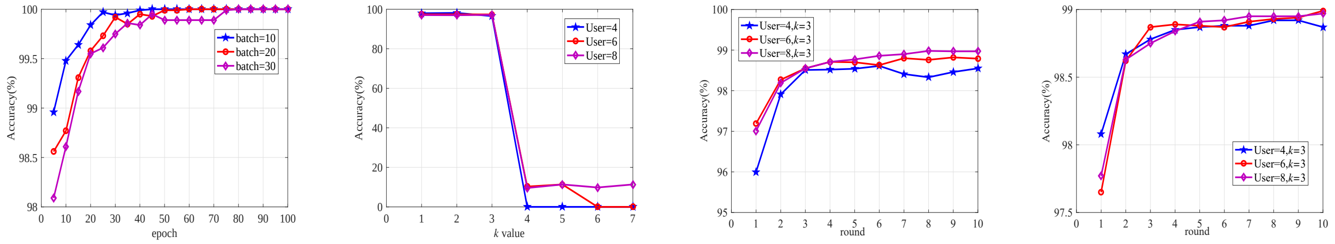
C. Accuracy

In the subsection, the accuracy of our scheme is tested without considering the privacy-preserving mechanism owing to the accuracy is not affected in plaintext or ciphertext. The experiment result is shown in Fig. 8. In Fig. 8(a), we compare the accuracy with epoch variation in different batches (10, 20, 30). The accuracy reaches 100% in epoch for (45, 60, 80), which corresponds to batch (10, 20, 30) respectively. The results show that the batch is 10, the accuracy is high. Then, we test the accuracy against k value choice based on KNN. As shown in Fig. 8(b), when k values are $\{1, 2, 3\}$ in different number of users $\{4, 6, 8\}$, the accuracy is high. So we choose 3 as k value to test the accuracy. As shown in Figs. 8(c), 8(d), we test the accuracy when the number of users is (4, 6, 8) and k is 3. The results demonstrate that the accuracy increases constantly with the round number $\{1, 2, \ldots, 10\}$ growing.



(a) Encryption time (b) Decryption time (c) Total time

Fig. 7. Performance analysis of cryptosystems



(a) Accuracy against epoch (b) Accuracy against k value choice (c) Accuracy against k value (d) Accuracy against k value

Fig. 8. Accuracy analysis

From Figs. 8(b), 8(c), 8(d), we can see that the appropriate choice of k value in OFFLF directly affects the correctness.

VIII. Conclusion

FL as a popular collaborative learning method, its privacy protection ways has been researched in many ways. However, in IoT or edge computing scenarios, the FL framework does not apply well owing to high-communication cost, low-quality data. To build a practical FL framework for IoT, we propose PFLEKA framework with MEEC, which introduces ad-hoc network and KNN method to FL, proposes an ad-hoc quit and join algorithm and Euclidean fraction method to improve a high-quality data application based on KNN method, reducing the communication and computational costs, solving the key bottleneck questions for IoT in aforementioned. The scheme is the first FL framework based on ad-hoc network and the KNN method. In future work, we will focus on parallel computing and incentive mechanisms in the privacy-preserving FL framework.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant Nos. U19B2021, 61972457, the Key Research and Development Program of Shaanxi under Grant No. 2020ZDLGY08-04, the Key Technologies R & D Program of Henan Province under Grant Nos. 212102210084 and 192102210295, and the Innovation Scientists and Technicians Troop Construction Projects of Henan Province.

References

[1] G. Xu, H. Li, Y. Zhang, S. Xu, J. Ning, and R. Deng, “Privacy-preserving federated deep learning with irregular users,” *IEEE Trans. Dependable Secur. Comput.*, vol. PP, no. 99, pp. 1–1, 2020.

[2] X. Ma, F. Zhang, X. Chen, and J. Shen, “Privacy preserving multi-party computation delegation for deep learning in cloud computing,” *Inf. Sci.*, vol. 459, pp. 103–116, 2018.

[3] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, “Efficient and privacy-enhanced federated learning for industrial artificial intelligence,” *IEEE Trans. Ind. Informatics*, vol. 16, no. 10, pp. 6532–6542, 2020.

[4] T. Li, J. Li, X. Chen, Z. Liu, W. Lou, and Y. T. Hou, “NPMML: A framework for non-interactive privacy-preserving multi-party machine learning,” *IEEE Trans. Dependable Secur. Comput.*, vol. 18, no. 6, pp. 2969–2982, 2021.

[5] R. Shokri and V. Shmatikov, “Privacy-preserving deep learning,” in *Proc. ACM SIGSAC*, 2015, pp. 1310–1321.

[6] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, “Privacy-preserving deep learning via additively homomorphic encryption,” *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 5, pp. 1333–1345, 2018.

[7] L. Zhao, Q. Wang, Q. Zou, Y. Zhang, and Y. Chen, “Privacy-preserving collaborative deep learning with unreliable participants,” *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 1486–1500, 2020.

[8] Y. Chen, Y. Ping, Z. Zhang, B. Wang, and S. He, “Privacy-preserving image multi-classification deep learning model in robot system of industrial iot,” *Neural Comput. Appl.*, vol. 33, no. 10, pp. 4677–4694, 2021.

[9] C. Dwork, “Differential privacy,” in *Proc ICALP 2006*, vol. 4052, 2006, pp. 1–12.

[10] L. Lyu, Y. Li, K. Nandakumar, J. Yu, and X. Ma, “How to democratise and protect AI: fair and differentially private decentralised deep learning,” *CoRR*, vol. abs/2007.09370, 2020.

[11] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, “A hybrid approach to privacy-preserving federated learning,” in *Proc. AISec@CCS 2019*, 2019, pp. 1–11.

[12] G. Yang, S. Wang, and H. Wang, “Federated learning with personalized local differential privacy,” in *Proc. ICCCS 2021*, 2021, pp. 484–489.

[13] Z. Xiong, Z. Cai, D. Takabi, and W. Li, “Privacy threat and defense for federated learning with non-i.i.d. data in aiot,” *IEEE Trans. Ind. Informatics*, vol. PP, no. 99, pp. 1–1, 2021.

[14] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, “Hybridalpha: An efficient approach for privacy-preserving federated learning,” in *Proc. AISec@CCS 2019*, 2019, pp. 13–23.

[15] J. Feng, L. T. Yang, Q. Zhu, and K. R. Choo, “Privacy-preserving tensor decomposition over encrypted data in a federated cloud environment,” *IEEE Trans. Dependable Secur. Comput.*, vol. 17, no. 4, pp. 857–868, 2020.

[16] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive laplace mechanism: Differential privacy preservation in deep learning," in *Proc. ICDM*, 2017, pp. 385–394.

[17] M. Gong, J. Feng, and Y. Xie, "Privacy-enhanced multi-party deep learning," *Neural Networks*, vol. 121, pp. 484–496, 2020.

[18] M. Abadi, A. Chu, I. J. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proc. ACM SIGSAC*, 2016, pp. 308–318.

[19] L. Xiang, J. Yang, and B. Li, "Differentially-private deep learning from an optimization perspective," in *Proc. INFOCOM 2019*, 2019, pp. 559–567.

[20] A. Bansal, T. Chen, and S. Zhong, "Privacy preserving back-propagation neural network learning over arbitrarily partitioned data," *Neural Comput. Appl.*, vol. 20, no. 1, pp. 143–150, 2011.

[21] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC*, 2017, pp. 1175–1191.

[22] X. Zhang, X. Chen, J. K. Liu, and Y. Xiang, "Deeppar and deepdpa: Privacy preserving and asynchronous deep learning for industrial iot," *IEEE Trans. Ind. Informatics*, vol. 16, no. 3, pp. 2081–2090, 2019.

[23] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Comput. Theory*, vol. 6, no. 3, pp. 13:1–13:36, 2014.

[24] B. Wang, Y. Zhan, and Z. Zhang, "Cryptanalysis of a symmetric fully homomorphic encryption scheme," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 6, pp. 1460–1467, 2018.

[25] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems (reprint)," *Commun. ACM*, vol. 26, no. 1, pp. 96–99, 1983.

[26] Y. Chen, B. Wang, and Z. Zhang, "Pdlhr: Privacy-preserving deep learning model with homomorphic re-encryption in robot system," *IEEE Systems Journal*, pp. 1–12, 2021.

[27] E. Bresson, D. Catalano, and D. Pointcheval, "A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications," in *Proc. ASIACRYPT 2003*, vol. 2894, 2003, pp. 37–54.

[28] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 31, no. 10, pp. 4229–4238, 2020.

[29] J. Mills, J. Hu, and G. Min, "Communication-efficient federated learning for wireless edge intelligence in iot," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5986–5994, 2020.

[30] A. Cotter, H. Jiang, M. R. Gupta, S. Wang, T. Narayan, S. You, and K. Sridharan, "Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals," *J. Mach. Learn. Res.*, vol. 20, pp. 172:1–172:59, 2019.

[31] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proc. ICML 2019*, vol. 97, 2019, pp. 4615–4625.

[32] J. Pang, Y. Huang, Z. Xie, Q. Han, and Z. Cai, "Realizing the heterogeneity: A self-organized federated learning framework for iot," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3088–3098, 2021.

[33] D. Bogdanov, S. Laur, and J. Willemson, "Sharemind: A framework for fast privacy-preserving computations," in *Proc. ESORICS 2008*, vol. 5283, 2008, pp. 192–206.

[34] Y. Zhang, G. Bai, X. Li, C. Curtis, C. Chen, and R. K. L. Ko, "Privcoll: Practical privacy-preserving collaborative machine learning," in *Proc. ESORICS 2020*, vol. 12308, 2020, pp. 399–418.

[35] N. Koblitz, A. Menezes, and S. A. Vanstone, "The state of elliptic curve cryptography," *Des. Codes Cryptogr.*, vol. 19, no. 2/3, pp. 173–193, 2000.

[36] N. Koblitz, *A course in number theory and cryptography, Second Edition*, ser. Graduate texts in mathematics. Springer, 1994, vol. 114.

[37] L. Li, A. A. A. El-Latif, and X. Niu, "Elliptic curve elgamal based homomorphic image encryption scheme for sharing secret images," *Signal Process.*, vol. 92, no. 4, pp. 1069–1078, 2012.

[38] Y. E. Housni, "Introduction to the mathematic foundations of elliptic curve cryptography," in *chapter III: Elliptic Curve Cryptography*, 2018, p. 18. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01914807

[39] M. Khammash, R. Tammam, A. Masri, and A. Awad, "Elliptic curve parameters optimization for lightweight cryptography in mobile-ad-hoc networks," in *Proc. SSD 2021*, 2021, pp. 63–69.

[40] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, vol. 353, 1994, pp. 153–181.

Yange Chen is currently learning for a Ph.D. in School of Telecommunications Engineering, Xidian University, and she is an associate professor in School of Information Engineering, Xuchang University. She received her MS and BS degrees in computer application technology from Henan Polytechnic University in 2008 and 2006, respectively. Her main research interests include deep learning security, Internet of Things security, homomorphic encryption.

Baocang Wang is a professor in the School of Telecommunications Engineering, Xidian University. He received his Ph.D. degree in cryptography from Xidian University in 2006, and received his MS and BS degrees in mathematics from Xidian University in 2004 and 2001, respectively. His main research interests include public key cryptography, encryption data processing, data mining security.

Rongxing Lu (S'09-M'11-SM'15-F'21) is currently an associate professor at the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada. He is a Fellow of IEEE. His research interests include applied cryptography, privacy enhancing technologies, and IoT-Big Data security and privacy. He has published extensively in his areas of expertise, and was the recipient of 9 best (student) paper awards from some reputable journals and conferences. Currently, Dr. Lu serves as the Vice-Chair (Conferences) of IEEE ComSoc CIS-TC (Communications and Information Security Technical Committee). Dr. Lu is the Winner of 2016-17 Excellence in Teaching Award, FCS, UNB.

Xu An Wang is a professor in Engineering University of People's Armed Police. His main research interests include cloud computing, cryptography, security and privacy for IoT, information security. He has published about 100 papers in the field of cloud computing, cryptography, information security and computer science. He is a Co-chair or TPC members of several international conferences like INCOS, 3PGCIC, EIDWT, FCS, CISIS etc. He has guest editor several special issues in international journals, he is an editor board member of several international journals like IJGUC, IJITWE, IJTHI etc.