scientific reports

Check for updates

OPEN

PQSF: post-quantum secure privacy-preserving federated learning

Xia Zhang[1], Haitao Deng[2], Rui Wu[2], Jingjing Ren[2] & Yongjun Ren[2]✉

In federated learning, secret sharing is a key technology to maintain the privacy of participants' local models. Moreover, with the rapid development of quantum computers, existing federated learning privacy protection schemes based on secret sharing will no longer be able to guarantee the data security of participants in the post-quantum era. In addition, existing privacy protection methods have the problem of high communication and computational overhead. Although the multi-stage secret sharing scheme proposed by Pilaram et al. is one of the effective solutions to the above problems, existing studies have proven the privacy leakage risk of this scheme. This paper firstly designs a new lattice-based multi-stage secret sharing scheme *Improved-Pilaram* to solve the security problem, which allows participants to use public vectors to reconstruct different secret values without changing the secret sharing. Based on *Improved-Pilaram*, this article proposes a post-quantum secure federated learning scheme *PQSF*. *PQSF* uses double masking technology to encrypt model parameters and achieves mask reconstruction through secret sharing. Since *Improved-Pilaram* is multi-stage, participants do not need to update their local secret shares frequently during training. Analysis and experimental results show that the *PQSF* proposed in this paper reduces the communication complexity between participants and reduces the computational overhead by about 20% compared with existing solutions.

Keywords Federated learning, Secure aggregation, Secret sharing, Post quantum security

In 2016, federated learning was first proposed as a distributed machine learning technology[1]. Its main idea is to perform distributed machine learning model training among multiple participants with local data sets . This technology can jointly train a global model by exchanging only the intermediate parameters of the training model without uploading local sample data[2]. Once the concept of federated learning was proposed, it has received widespread attention from experts in the industry and academia as a key technology to break "data silos" and realize data mining[3,4].

However, as research continues to deepen, the privacy protection capabilities of federated learning face severe challenges[5]. With the emergence of reverse attacks and inference attacks, users' local sensitive data may be leaked due to unencrypted model parameters[6]. While enhancing the security of federated learning, providing privacy protection functions has become one of the focuses of current research. Currently, privacy protection methods for federated learning are mainly implemented through three technologies: Differential Privacy (DP), Homomorphic Encryption (HE) and Secure Multi-Party Computation (SMPC). Differential privacy is a technology that protects privacy by adding a certain amount of random noise to the data[7]. Homomorphic encryption protects data privacy by directly calculating ciphertext[8]. However, aggregation methods based on differential privacy will destroy the accuracy of federated learning training results, while homomorphic encryption will greatly increase the computational overhead of federated learning. In comparison, privacy-preserving protocols based on secure multi-party computation generally have lower computational overhead and higher computational accuracy[9,10].

As one of the underlying technologies of secure multi-party computation, secret sharing is widely used in federated learning to achieve user privacy protection[11,12]. Currently, with the rapid development of quantum computing technology, the security of traditional secret sharing schemes can no longer be guaranteed, and post-quantum secure secret sharing technology is becoming one of the hot topics of current research[13,14]. As an important candidate scheme, lattice cipher has attracted widespread attention due to its efficient computing

[1]Jiangsu Collaborative Innovation Center of Chinese Medicinal Resources Industrialization, School of Artificial Intelligence and Information Technology, Nanjing University of Chinese Medicine, Nanjing 210023, China. [2]School of Computer Science, School of Cyber Science and Engineering, Nanjing University of Information Science and Technology, Nanjing 210044, China. ✉email: renyj100@126.com

power and versatility. Multi-stage secret sharing can reconstruct and recover different secrets, so it is very suitable for federated learning scenarios. In 2017, Pilaram et al. proposed a lattice-based multi-stage secret sharing scheme[15]. This scheme can perform secret reconstruction through shares and public parameters without exposing secret share information during the multi-stage reconstruction process. However, the scheme they proposed was proven to have security risks in 2023[16]. When the number of recovered secrets multiplied by the threshold value is greater than the share length, the security of the remaining unrecovered secrets will be destroyed. This threat will be further amplified in a federated learning environment.

Privacy protection has received widespread attention as an important function in federated learning. However, existing privacy protection schemes based on secret sharing require complex computation and communication overhead among participants. On the other hand, the rapid development of quantum computers is threatening the security of traditional cryptography. How to build a quantum-resistant and communication-efficient federated learning based on a multi-stage secret sharing scheme is the focus of this article. Therefore, we investigate in detail existing federated learning privacy-preserving methods as well as post-quantum secure multi-stage secret sharing techniques. In view of the risk of privacy leakage described by Yang et al.[16], the scheme[15] was improved and a secure multi-stage secret sharing scheme *Improved-Pilaram* was proposed. Based on *Improved-Pilaram*, a post-quantum secure federated learning security aggregation scheme *PQSF* is designed. The main contributions of this article are as follows:

(1) First, the existing lattice-based multi-stage secret sharing scheme[15] is improved, and a secure secret sharing protocol *Improved-Pilaram* is proposed. This protocol enables the same secret share to participate in reconstructing multiple secret messages without destroying the security of other unreconstructed secrets.
(2) Based on *Improved-Pilaram*, this paper designs a post-quantum secure federated learning privacy protection scheme *PQSF* combined with double mask technology. Due to the multi-stage nature of *Improved-Pilaram*, training participants do not need to frequently update local shares during the federated learning process. Instead, they use different public parameters to perform secret reconstruction, thereby effectively reducing the communication complexity between participants.
(3) This paper conducts security analysis on the proposed secret sharing protocol and federated learning scheme, and conducts simulation experiments. This article compares *PQSF* with the existing post-quantum secure federated learning scheme LaF[17]. Analysis and experimental results show that the post-quantum secure privacy-preserving federated learning *PQSF* based on *Improved-Pilaram* effectively reduces the computing overhead by about 20%.The rest of this paper is organized as follows: First, section "Releated work" of the paper briefly introduces related research work in the field of privacy-preserving federated learning and secret sharing. Then we briefly introduce the relevant technical basis used in this paper in section "Preliminary". The proposed lattice-based multi-stage secret sharing scheme *Improved-Pilaram* is introduced in detail in section "Improved-pilaram: lattice-based multi-stage secret sharing scheme", which includes the detailed construction and security analysis of the scheme. Then section "The proposed PQSF" introduces the detailed execution process and privacy protection method of the proposed post-quantum secure federated learning protocol *PQSF*. In section "Simulation experiment and analysis", the performance of the proposed scheme is analyzed through simulation experiments and compared with previous schemes. Concluding observations are found in section "Conclusion".

Releated work

Privacy-preserving federated learning

In 2016, the Google AI team[18] proposed a distributed machine learning[19] framework-federated learning, which was subsequently experimented with on mobile device input prediction and achieved success. Its core involves training machine learning models on user devices, completing multi-party aggregation through a server, and updating global parameters, a process that cycles continuously until training objectives or termination conditions are reached. However, the method of achieving global model training by sharing local parameter information is not sufficient to protect data privacy. For example, Song et al.[20] introduced mGAN-AI to analyze privacy leakage issues in federated learning, and they were the first to achieve user-level privacy leakage through attacks by malicious servers. Currently, privacy protection methods for federated learning are mainly implemented through three technologies: Differential Privacy, Homomorphic Encryption, and Secure Multi-Party Computation[21].

Differential privacy is a technique that protects privacy by adding a certain amount of random noise to the data[7]. Zhao et al. proposed a federated learning framework based on local differential privacy[22], which ensures security by deploying noise to model parameters. However, adding noise locally will be diluted in the process of averaging the global model, and the approach of achieving privacy protection through DP will inevitably affect the accuracy of the federated learning model. Bin et al.[23] proposed protecting the model through HE. In the proposed privacy protection scheme, a trusted terminal generates a public-private key pair for each participant. Each terminal encrypts local model parameters using the public key, and the central server computes the global model directly from the ciphertext. However, each participant in the model has the same private key, which cannot resist malicious terminal devices. Meanwhile, HE technology is computationally expensive and not suitable for lightweight, multi-participant federated learning frameworks.

Compared to the accuracy loss brought by DP and the high computational cost of HE, privacy protection protocols based on SMPC generally have lower computational overhead and higher computational accuracy[9]. Among them, secret sharing, as a classical cryptographic technique, is widely applied in the field of secure multi-party computation. Bonawitz et al.[24] proposed a privacy-protective secure aggregation scheme based on a participant-server framework in 2017. They protected the participants' model parameters through a double masking method and used the Shamir secret sharing algorithm to recover the mask information of offline participants. Inspired by their work, Duan et al.[25] used secret sharing technology to share gradient shares among

participants and upload them to the server after aggregation. However, with the advent of the post-quantum era, many existing schemes will no longer be secure, such as the Diffie–Hellman key agreement protocol based on the Discrete Logarithm Problem (DLP). To address this problem, Xu et al.[17], based on the work of Bonawitz et al., constructed a communication-efficient federated learning protocol LaF based on lattices. This scheme not only achieves post-quantum security but also avoids distributing new shares to all participants in each round of federated learning, saving a significant amount of communication overhead.

Secret sharing

Secret sharing was first introduced by Shamir[26] and Blakley[27]. Blakley constructed a threshold secret sharing scheme using hyper-geometric problems, while Shamir developed a secret sharing scheme using polynomial problems over finite fields. Secret sharing is commonly applied in scenarios such as key management and access control[28]. Furthermore, as a crucial technology for constructing secure multi-party computation schemes[29], secret sharing is also widely used in the field of privacy-preserving federated learning.

Ordinary secret sharing schemes cannot adapt well to various complex situations. For this reason, a large number of scholars have not only researched the construction of threshold sharing schemes using different tools but have also extensively studied the construction of threshold schemes with additional functionalities. The purpose of Verifiable Secret Sharing (VSS) is to make secret sharing robust against malicious parties. In 2020, Kandar et al. proposed a (t, n) VSS scheme[30] with aggregator verification and cheater detection. The scheme not only achieved verification of secret reconstruction correctness but also could detect participants who submitted incorrect shares. The development of this direction enables traditional secret sharing schemes to resist dishonest participants, effectively preventing mutual deception among distributors, participants, and between participants[31].

However, the underlying hard problems used to construct verifiable secret sharing schemes can no longer resist attacks from quantum computers, and the security of other schemes will also be threatened. In 1994, Shor[32] proposed a quantum algorithm that solves the factorization problem in polynomial time, and subsequently, more and more scholars have presented quantum cracking algorithm studies targeting numerical assumptions, demonstrating the vulnerability of schemes based on numerical assumptions[33]. As such, with the continuous maturation and improvement of quantum technology, the security of traditional secret sharing schemes will suffer greatly, and there is an urgent need to research new candidate schemes for quantum-resistant secret sharing.

In 2019, Rajabi et al. proposed a lattice-based verifiable secret sharing scheme based on the Shamir secret sharing scheme[34]. In this paper, the authors first proposed a general threshold verifiable secret sharing structure. The proposed scheme requires a set of collision-resistant homomorphic hash functions to verify shares and uses a Generalized Compact Knapsack (GCK) function to construct a lattice-based verifiable secret sharing scheme. To support large committees comprising thousands of participants, the scheme's communication and computation need to be sufficiently efficient. For this purpose, Gentry et al., based on the learning with errors problem, proposed a non-interactive publicly verifiable secret sharing scheme[35]. In 2017, Pilaram et al.[15] designed a multi-stage secret sharing scheme based on the Ajtai one-way function, and this scheme had multiple uses and was verifiable. However, this scheme was proven to have privacy leakage risks in 2023[16].

Preliminary

Lattice cryptography

Lattice cryptography is an encryption method based on lattice theory, which has received widespread attention in recent years due to its potential to resist quantum computing attacks, and is considered one of the most promising quantum-resistant cryptographic technologies[36]. A lattice is a set of linearly independent nonzero vectors and their integer linear combinations, with this set of linearly independent vectors known as the lattice basis. It is worth noting that the lattice basis of a lattice is not unique.

Definition 1 Given a set of linearly independent vectors $b_1, b_2, \ldots, b_n \in \mathbb{R}^m$, the lattice $L(B)$ generated by these vectors can be defined as:

$$L(B) = L(b_1, b_2, \ldots, b_n) = \left\{ \sum x_i b_i | x_i \in \mathbb{Z} \right\} \tag{1}$$

The vectors $(b_1, b_2, \ldots, b_n)$ are called a set of bases for the lattice $L(B)$, and B is defined as an $m \times n$ matrix whose column vectors are $b_1, b_2, \ldots, b_n$, then the lattice generated by the matrix can also be defined as:

$$L(B) = L(b_1, b_2, \ldots, b_n) = \{Bx | x \in \mathbb{Z}^n\} \tag{2}$$

Wherein, both m and n are integers, and $m \geq n$, m is called the dimension of the lattice, n is called the rank of the lattice, and a lattice that satisfies $m = n$ is called full rank. The security of lattice cryptography is premised on the difficulty of solving computational problems on lattices, such as the Shortest Vector Problem (SVP), the Closest Vector Problem (CVP), and Learning With Errors (LWE)[37]. These problems are exceedingly challenging to resolve within high-dimensional lattices, and this holds true even in the context of quantum computing.

Threshold secret sharing

A (t, n) threshold secret sharing scheme refers to a scenario where n users participate in secret sharing, and each participant obtains a secret share through the sharing algorithm. The original secret can only be reconstructed if a specific access structure is satisfied, that is, any t or more than t users participate in the secret reconstruction.

Any fewer than *t* participants cannot obtain any information about the secret *S*. Here, *t* is the threshold value. A threshold secret sharing mechanism mainly includes two stages: one is the share generation stage, and the other is the secret reconstruction stage. Suppose there are *n* participants in a secret sharing, along with a dealer Dealer and a secret *S*. The secret sharing algorithm is defined as follows:

Definition 2 If a mechanism includes a share generation process and a secret reconstruction process, and satisfies both security and correctness, then it is referred to as a threshold secret sharing algorithm.

Traditional secret sharing schemes generally consist of the following two stages:

(1) SS.Share(S,t,n):Inputs a secret S, a threshold value t, and the number of participants n. There exists a secret sharing algorithm that splits S into n secret shares:

$$SS.Share(S,t,n) \rightarrow \{s_1, s_2, \ldots, s_n\} \tag{3}$$

(2) SS.Recon($\{s_1, s_2, \ldots, s_n\}, t$):Input at least t secret shares, there exists a secret reconstruction algorithm that allows the original secret S to be reconstructed:

$$SS.Recon\{s_1, s_2, \ldots, s_n\} \rightarrow S \tag{4}$$

The secret sharing algorithm, while implementing share computation and reconstruction, also needs to satisfy the following security and correctness requirements:

(1) Security: When the number of known secret shares does not meet the threshold *t*, no participant can obtain information about the secret *S*. That is to say, even if multiple participants collaborate to reconstruct the original secret *S*, as long as the agreed access structure is not satisfied, the scheme can still guarantee the security of *S*.
(2) Correctness: When all participants execute the protocol according to the predetermined rules, no single party can reconstruct the secret on their own, but the collective of all authorized participants can accurately and correctly recover the original secret *S*.

Improved-Pilaram: lattice-based multi-stage secret sharing scheme

The secret sharing protocol *Improved-Pilaram* proposed in this section achieves multi-stage secret reconstruction without compromising the security of sharing by allowing the same secret share to participate in the reconstruction of multiple secret messages. The protocol uses shared and public parameters in each reconstruction stage to ensure that the secret information remains confidential during the reconstruction process. In this way, the *Improved-Pilaram* multi-stage secret sharing scheme achieves post-quantum security by utilizing lattice-based encryption and multilinear mapping techniques, ensuring the protection of secret information even in the presence of quantum computing threats. By partitioning secrets across multiple stages, the scheme minimizes the impact of potential leakage at any single stage on the overall confidentiality, thereby strengthening data security. Moreover, the multi-stage design complicates and raises the cost of attacks, effectively countering malicious participant actions and enhancing the overall security of the system.

Scheme construction

The lattice-based *Improved-Pilaram* algorithm mainly includes the following stages:

(1) *SS.Setup*(*v*, *t*): To share a secret *S*, the Dealer randomly selects a vector $v \in \mathbb{Z}_q^t$, where the last entry of vector *v* is 1, *v* is a prime number, and *t* is the threshold value of the *Improved-Pilaram*. Subsequently, the Dealer calculates a lattice basis *B* of dimension *t* for the secret *S*.

$$S = Bv \tag{5}$$

Where $B \in \mathbb{Z}_q^{t\times t}$. However, in this process, the solution to the equation is not unique, which could compromise the correctness of the *Improved-Pilaram* in the reconstruction phase. To solve this problem, the Dealer needs to split the basis *B* into $B_1 \in \mathbb{Z}_q^{t\times(t-1)}$ and $b \in \mathbb{Z}_q^t$, to achieve the correctness of Eq. (6).

$$\begin{aligned} S = Bv = [B_1, b]\begin{bmatrix} v' \\ 1 \end{bmatrix} \\ \Rightarrow b = S - B_1 v' \end{aligned} \tag{6}$$

Where v' is the first $t-1$ elements of the random vector *v* and the last element is 1, B_1 is randomly selected in $\mathbb{Z}_q^{t\times(t-1)}$, and finally, the Dealer publicizes the vector *v*.
(2) *SS.Share*(*S*, *t*, *n*): As shown in Fig. 1, in the secret share generation stage,set the secret S,the threshold t ,the Dealer selects *n* vectors $\lambda_j \in \mathbb{Z}_q^t, j = 1, 2, \ldots, n$, ensuring that every set of *t* vectors are linearly independent

of each other. The matrix $\Lambda = [\lambda_1, \ldots, \lambda_n] \in \mathbb{Z}_q^{t\times n}$ is the right product of any $t \times t$ random invertible matrix and a random Vandermonde matrix. Next, the Dealer needs to find a matrix $A \in \mathbb{Z}_q^{t\times r}$, a secret matrix $C = [c_1, \ldots, c_n] \in \mathbb{Z}_q^{r\times n}$ and vectors $e_j \in \mathbb{Z}_q^t, j = 1, 2, \ldots, n$, where $r \geq max(tlogt, n)$. To satisfy Eq. (7).

$$\begin{aligned} & AC + H(E) = B\Lambda \\ & \Rightarrow A\left[c_1, \ldots, c_n\right] + H(E) = B\left[\lambda_1, \ldots, \lambda_n\right] \\ & \Rightarrow \begin{cases} Ac_1 + H\left(e_1\right) = B\lambda_1 \\ Ac_2 + H\left(e_2\right) = B\lambda_2 \\ \quad\vdots \\ Ac_n + H\left(e_n\right) = B\lambda_n \end{cases} \end{aligned} \tag{7}$$

Where $H(\cdot)$ is a random permutation, E is a $t \times n$ order matrix composed of $e_j, j = 1, 2, \ldots, n$. Similar to the computation process of lattice basis *B*, the solution for the public matrix *A* is not unique. Therefore, the matrix *A* needs to be split into two unknown matrices $A_1 \in \mathbb{Z}_q^{t\times n}$,$A_2 \in \mathbb{Z}_q^{t\times(r-n)}n$, and the secret matrix C split into $C_1 \in \mathbb{Z}_q^{n\times n}$ and $C_2 \in \mathbb{Z}_q^{(r-n)\times n}$ to calculate a specific value. The Dealer first randomly selects and determines the value of A_2 in $\mathbb{Z}_q^{t\times(r-n)}$ and calculates A_1 through Eq. (8).

$$\begin{aligned} & AC + H(E) = B\Lambda \\ & \Rightarrow [A_1, A_2]\begin{bmatrix} C_1 \\ C_2 \end{bmatrix} + H(E) = B\Lambda \\ & \Rightarrow A_1 = \left(B\Lambda - H(E) - A_2C_2\right)C_1^{-1} \end{aligned} \tag{8}$$

Finally, the Dealer packages each column $c_j, j = 1, 2, \ldots, n$ and $e_j, j = 1, 2, \ldots, n$ of the secret matrix and sends it as a secret share $(c_j, e_j), j = 1, 2, \ldots, n$ to each *Improved-Pilaram* participant P_j through an encrypted channel, and discloses the matrix *A*,*C* and random arrangement method $H(\cdot)$.

(3) $SS.Recon(\{s_1, s_2, \ldots, s_n\}, t)$: As shown in Fig. 2, in the secret reconstruction phase, assume there exists a threshold number *t* of participants$P_i, i \in 1, 2, \ldots, t$ attempting to collaboratively reconstruct the secret *S*. Each participant locally uses their secret shares (c_i, e_i), the public matrix A, and the random permutation $H(\cdot)$ to calculate the secret information $msg_i = Ac_i + H(e_i)$ and uploads it. The secret reconstruction initiator, after receiving a sufficient amount of secret information, selects the corresponding t column vectors from the Λ matrix for participants P_i to obtain Λ_{P_i}, and finally calculates the lattice basis *B* for the secret *S*.

$$\begin{aligned} & AC_{P_i} + H(E)_{P_i} = B\Lambda_{P_i} \\ & \Rightarrow B = (AC_{P_i} + H(E)_{P_i})\Lambda_{P_i}^{-1} \end{aligned} \tag{9}$$

Based on Eq. (5), the secret aggregation initiator uses the calculated lattice basis *B* and the public vector *v* to reconstruct the original secret $S = Bv$.

(4) $SS.Renew(S', t, n)$: In the secret update stage, to share a new secret S', the Dealer executes the initialization phase again with the new secret S' to obtain the lattice basis B'. In the new *Improved-Pilaram* stage, the original secret shares $(c_j, e_j), j = 1, 2, \ldots, n$ remain unchanged, and calculate:

$$A'C + H'(E) = B'\Lambda \tag{10}$$

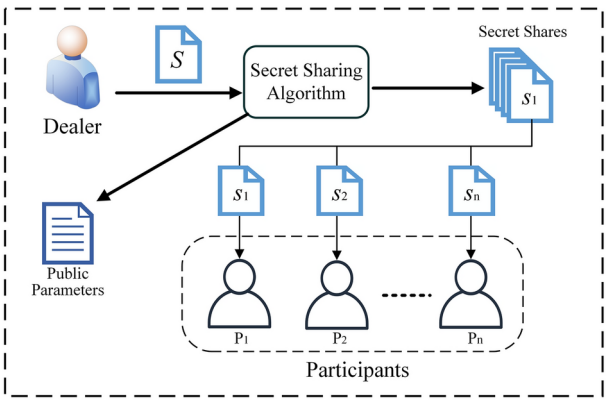

Figure 1. Secret sharing stage.

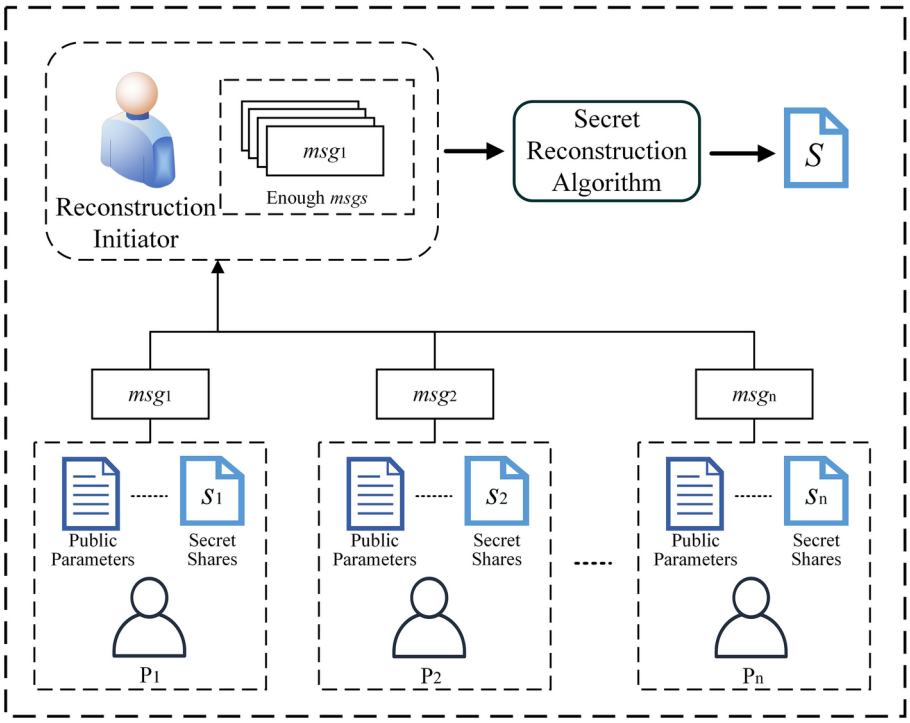

Figure 2. Secret reconstruction stage.

Finally, the public matrix A' and the random permutation method $H'(\cdot)$, as new secret reconstruction parameters, are publicized. During the sharing process of the new secret S', the secret sharing shares of participant P_j do not need to change; instead, secret reconstruction is achieved by updating the public parameters, completing a new round of secret sharing operations. This approach greatly reduces the communication overhead and complexity between participants.

Compared with traditional federated learning privacy protection methods, the PQSF scheme is based on the improved multi-stage secret sharing scheme Improved-Pilaram, which can reconstruct secrets using different public parameters without frequently updating local secret shares, thereby reducing computational and communication overheads.

Security analysis

Correctness

The correctness of this *Improved-Pilaram* requires satisfying two conditions. First, after obtaining secret shares and public parameters through the initial execution of the sharing algorithm, during the secret reconstruction phase, at least t participating parties holding shares should be able to cooperatively use the secret shares and public parameters to reconstruct the original secret. Second, after performing secret updating to share a new secret, using the original secret shares and new public parameters, under the same premise of cooperation from at least t participating parties, the new round of secret information should be correctly recovered.

Theorem 1 *For any secret S, threshold t, and number of participating parties* $n(1 \leq t \leq n)$*, using public parameters* $(A, \Lambda, v, H())$ *and participating parties* $P_i, i = 1, 2, \ldots, t$ *to compute secret information* $Ac_i + H(e_i)$*, the secret recovery algorithm can correctly restore secret S when the number of shares* $n \geq t$*.*

Proof During the secret reconstruction phase, participating parties $P_i, i = 1, 2, \ldots, t$ calculate $Ac_i + H(e_i)$ using share information (c_i, e_i) and public parameters A and random permutation $H(\cdot)$, and upload them. The initiator of the secret reconstruction constructs Λ_{P_i} and $AC_{P_i} + H(E)_{P_i}$ based on the information from participating parties to satisfy Eq. (11):

$$(AC_{P_i} + H(E)_{P_i})\Lambda_{P_i}^{-1} = B\Lambda_{P_i}\Lambda_{P_i}^{-1} = B \tag{11}$$

Finally, according to Eq. (5), multiplying the lattice basis B obtained by the reconstruction algorithm with the public parameter vector v, the secret S can be correctly reconstructed. Theorem 1 is proved.

Theorem 2 *For secret* S'*, threshold t, and number of participating parties* $n(1 \leq t \leq n)$*, using public parameters* $(A', \Lambda, v', H'(\cdot))$ *and participating parties* $P_i, i = 1, 2, \ldots, t$*, to compute secret information* $A'c_i + H'(e_i)$*, the secret recovery algorithm can correctly restore the new round of secret* S' *when the number of shares* $n \geq t$*.*

Proof During the secret updating phase, the Dealer calculates lattice basis B' and public parameters A' and $H'(\cdot)$ using the new secret S'. By analogy with Theorem 1, the correctness of Theorem 2 can be proved.

Privacy protection
The security of the *Improved-Pilaram* proposed in this section needs to satisfy several conditions. First, in the proposed scheme, any subset of participating parties with fewer than t members should not be able to reconstruct the original secret *S*. Second, during the multi-stage secret sharing process, malicious participants should not be able to use known public vector information from previous rounds to compromise the security of secrets in the latest round.

Theorem 3 *For any secret S, givenmsecret shares* (c_j, e_j) *and public parameters* $(A, \Lambda, v, H(\cdot))$*, when the number of shares* $m < t$*, the original secret S cannot be reconstructed.*

Proof Assume in the worst-case scenario, $t-1$ participants $P_k, k = 1, 2, \ldots, t-1$, conspire to attempt to compromise the security of secret *S* through the reconstruction algorithm. They first calculate $AC_{P_k} + H(E)_{P_k}$ and Λ_{P_k} using their own secret shares and public parameters and obtain a B_k.

$$B_k = (AC_{P_k} + H(E)_{P_k})\Lambda_{P_k}^{-1} \tag{12}$$

However, B_k is a $t \times (t-1)$ dimensional matrix that cannot reconstruct secret *S* through Eq. (5). Therefore, when the number of shares $m < t$, the value of original secret *S* cannot be reconstructed. Theorem 3 is proved.

Theorem 4 *Suppose the currently shared secret is the x-th secret* S_x*, given reconstructed secrets* $S_i, i = 1, 2, \ldots, x-1$ *, and public parameters* $(A_l, \Lambda, v, H_l(\cdot)), l = 1, 2, \ldots, x$*, malicious conspiring participants cannot compromise the security of the latest secret* S_x *using this information.*

Proof It is known from the *Improved-Pilaram* algorithm that participating party P_j computes $A_x c_j$ based on secret share c_j, which is uniformly distributed over $\mathbb{Z}_q^t$. Furthermore, since the permutation method $H(\cdot)$ used is also random, $H_x(e_j)$ is also uniformly distributed over $\mathbb{Z}_q^t$. It can be inferred that $A_x c_j + H_x(e_j)$ obtained from secret shares (c_j, e_j) also satisfies the characteristic of uniform distribution. Therefore, attackers find it difficult to obtain useful information about secret S_x from $A_x c_j + H_x(e_j)_{j=1}^n$. In the secret updating phase, the Dealer calculates and publishes new public parameters based on secret S_x, which also satisfy the characteristic of uniform distribution. As long as the permutation method $H(\cdot)$ used is random, the scheme remains secure. Therefore, the privacy of unreconstructed secrets is not compromised by reconstructed secrets. Theorem 4 is proved.

The proposed PQSF

In this section, we propose a *PQSF* aggregation scheme based on the secret sharing scheme presented in section "Improved-pilaram: lattice-based multi-stage secret sharing scheme", combined with a dual-mask mechanism. This method utilizes lattice-based key exchange protocols for Key Agreement among participating entities and employs dual masking to encrypt model gradients, thereby achieving privacy protection. Finally, the central server and online training participants reconstruct the dual masks through a *Improved-Pilaram* algorithm and ultimately update the global model.

Overview of the PQSF

System model
The goal of the scheme proposed in this paper is to use lattice-based multi-stage *Improved-Pilaram* technology to build a secure aggregation protocol, thereby providing a privacy-protected federated learning system that offers post-quantum security while reducing communication overhead between participants. We utilize a lattice-based key agreement scheme to generate masks between participants and protect the local training model gradients using a double-masking method. The system model of this scheme, as shown in Fig. 3, includes a central server and multiple training participants:

(1) Training Participants: Training participants, the local data owners in federated learning, refer to the entities that participate in training using their local data and upload models for aggregation. To protect the security of local data, data owners need to perform key agreement to generate masks that protect the model. Additionally, to prevent aggregation errors caused by participants dropping out during federated learning training, data owners also need to execute a *Improved-Pilaram* algorithm to reconstruct masks.
(2) Central Server: The central server, the main body completing model aggregation tasks in federated learning, typically has strong storage and computational capabilities. During the federated learning training process, the server first generates initial global model parameters and sends them to participants for training. In each training round, the server needs to use an aggregation algorithm to iteratively update the global model and collect secret information to restore the mask information of participants who have dropped out.

Threat model and design goals
In the *PQSF* proposed in this paper, we assume a semi-honest model. Training participants correctly execute the protocol for federated learning training, generate symmetric public and private keys through the key agreement protocol, and encrypt gradients using a lattice-based secret sharing algorithm combined with double-masking. The central server correctly forwards messages between participants, completes gradient aggregation, and

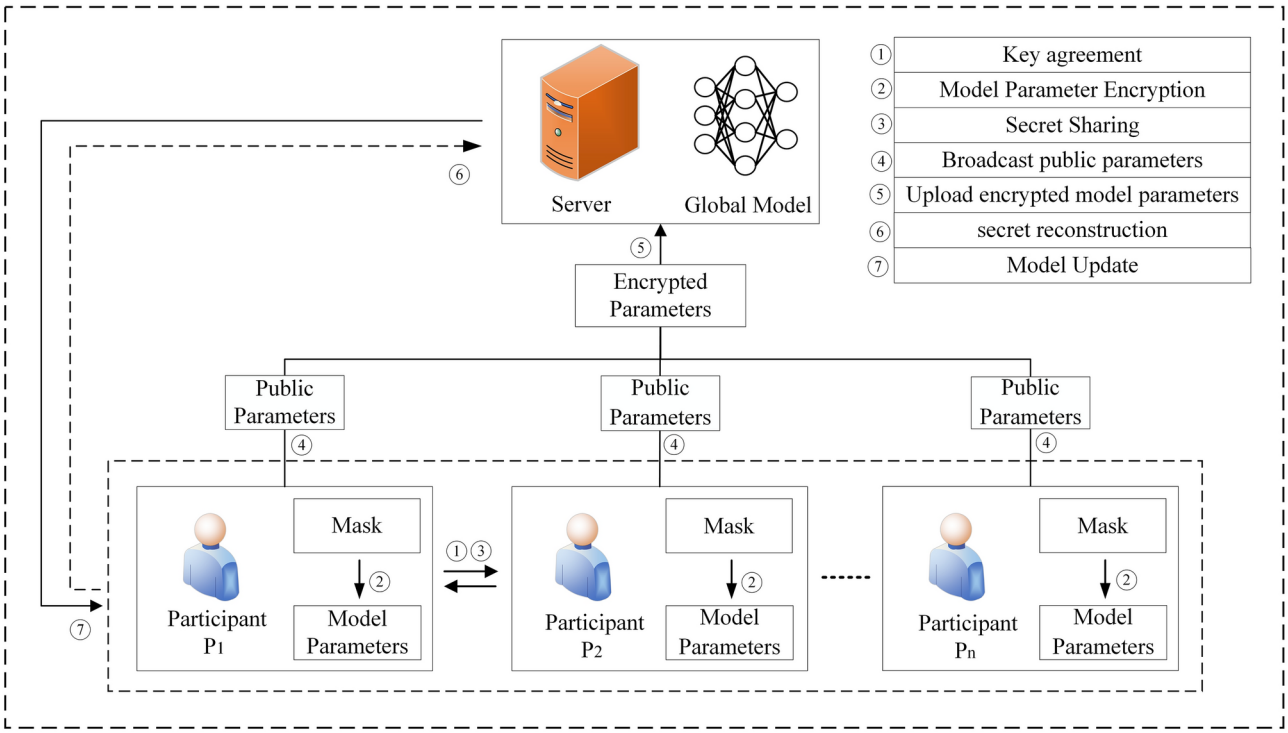

Figure 3. *PQSF* secure aggregation model.

eliminates the masks of participants who have dropped out through secret reconstruction. However, they will try to collect information during the process to infer other participants' private data or model details, thereby extracting as much knowledge as possible.

Based on the above threat model, to reduce communication complexity and protect data privacy, our proposed non-interactive federated learning secure aggregation scheme should meet privacy protection, post-quantum security, effectiveness, and communication efficiency. The specific scheme requirements are as follows:

(1) Privacy Protection: Privacy protection means that during the training process of the scheme, no adversary can learn any effective information about other participants' local datasets, ensuring the security of participants' local datasets. In the proposed scheme, besides the aggregated result, the training server cannot obtain any other information about the model parameters. At the same time, honest but curious participants also cannot learn other participants' local model parameters.
(2) Post-Quantum Security: Post-quantum security means that even with quantum computing capabilities, attackers cannot crack or infer the content of training participants' local datasets. Privacy protection schemes designed based on anti-quantum secure algorithms can ensure data's post-quantum security.
(3) Effectiveness: Effectiveness means that the gradient aggregation value calculated through the privacy protection method is correct. Following the protocol to execute gradient encryption and decryption leads to correct computation results, and the correctness of the aggregated model is not compromised due to participants dropping out.
(4) Communication Efficiency: Communication efficiency means that the proposed federated learning model aggregation scheme can significantly reduce the amount of data transmission required while protecting the privacy of training participants.

Our construction

Initial stage

Before the formal training begins, the central server initializes the model and parameters. Model initialization involves setting the parameters of the original federated learning neural network model, including learning rate and training epochs, denoted as w^{init}, and sending them to all participating data owners. Parameter initialization refers to the central server setting reasonable system security parameters such as k, a large prime number q, the number of participants n, the secret sharing threshold value t, and the pseudorandom number generator $PRG(\cdot)$, based on the participating data owners.

Local training and gradient encryption stage

In the local training phase, the model gradient is encrypted through a double mask mechanism. Before the participant uploads the gradient, the system first exchanges keys through the lattice-based NewHope protocol to generate a mask for gradient encryption. This key exchange mechanism not only reduces the complex communication steps of frequent key generation and distribution in each round of training, but also ensures that

in the event of a participant disconnection, the remaining participants can correctly reconstruct the required keys and masks through the shared public parameters, thereby successfully completing model aggregation. The following is an introduction to the specific solution:

(1) Local Training: In the *k*-th round of local training, data owner P_i updates the local model using the global gradient information w^k and trains it with dataset D_i to obtain new gradient parameters w_i^{k+1}. In traditional federated learning algorithms, the server aggregates the gradient information of participants to update the global gradient. However, directly uploading the gradient parameters w_j^{k+1} of participants may compromise the privacy of local datasets due to potential model inference attacks. Therefore, this protocol protects data privacy by adding masks before uploading participant gradients.

(2) Key Agreement: The dual-mask protocol used in this scheme is designed based on the key negotiation scheme. Before encryption, each participant interacts with other participants through the server. Compared with traditional key exchange methods, the NewHope protocol has quantum resistance, high computational efficiency, moderate key size and low communication overhead, ensuring that it can continue to operate safely even when participants are offline, providing solid security for the post-quantum era. This scheme uses the NewHope protocol to negotiate and calculate two symmetric keys between participants P_j and P_i : $key_{i,j}^c$ and $key_{i,j}^s$. Here, $key_{i,j}^c$ is used for encrypting messages between the two participants, while $key_{i,j}^s$ is used for encrypting gradient information during model aggregation. It is worth noting that if the current number of online participants is less than the threshold value $t(|U_1| < t)$, the protocol is terminated during server message forwarding.

(3) Gradient Encryption: During the gradient encryption phase, gradient masks are calculated using the symmetric key $key_{i,j}^s$ between data owners. Specifically, each participating online data owner P_i first determines whether each of the other participants P_j satisfies $i < j$ and calculates the specific values of the masks using a pseudorandom number generator $PRG(\cdot)$. Finally, the mask information is used to encrypt the local model w_i^{k+1} of each participating data owner P_i as follows:

$$y_i^{k+1} = w_i^{k+1} + \sum_{j \in U, j<i} PRG\left(key_{i,j}^s\right) - \sum_{j \in U, j>i} PRG\left(key_{i,j}^s\right) \tag{13}$$

Based on the properties of negotiated keys and pseudorandom number generators, all model gradients are mutually offset during server upload and final aggregation, resulting in correct aggregation results.

$$PRG\left(key_{i,j}^s\right) - PRG\left(k^s y_{i,j}^s\right) = 0 \tag{14}$$

However, there is a security risk in this scheme if a participant drops out or disconnects midway, potentially exposing the original gradient information. To address this, an additional random mask is added to the single-mask scheme. In this process, participant P_i randomly samples a seed b_i and modifies the single-mask scheme to a dual-mask scheme as follows:

$$y_i^{k+1} = w_i^{k+1} + PRG\left(b_i\right) + \sum_{j \in U, i<j} PRG\left(key_{i,j}^s\right) - \sum_{j \in U, i>j} PRG\left(key_{i,j}^s\right) \tag{15}$$

After the central server completes model aggregation, the privacy of model gradients is still protected by the randomness of $PRG(b_i)$, thereby resisting inference attacks.

During the gradient encryption process, each participant applies a static mask and a random dynamic mask to double-mask the gradients. This method ensures that even if the dynamic mask of a certain round is intercepted by the attacker, the attacker cannot infer the true gradients of other rounds since the dynamic mask changes randomly in each round of training. By introducing random dynamic masks, the unpredictability of the masks in each round of training significantly enhances privacy protection and prevents attackers from cracking the model gradient through fixed patterns or repeated attacks, thereby effectively improving privacy security in federated learning.

Compared with traditional federated learning privacy protection methods, PQSF utilizes a double mask mechanism to protect model gradients, ensuring that model gradients can be correctly aggregated during the encrypted upload process even when participants are offline. This mechanism may increase certain computational overhead in the initial stage, but in the subsequent training stage, since there is no need to frequently generate and update secret shares, the overall computational overhead will be significantly reduced.

(4) Secret Sharing: During the federated learning training process, the dropout of participants can lead to the inability of gradient masks to correctly offset each other, thereby compromising the correctness of the model aggregation results. Therefore, to eliminate random numbers and dropped participant mask information after model aggregation to obtain correct results, training participantsP_i convert the random seeds of the symmetric key $key_{i,j}^s$ into elements $seed_i^{sk}$ and $seed_i^e$ on $\mathbb{Z}_q^t$ and run the secret sharing algorithm SS.share from section "Preliminary" to compute b_i and the corresponding shares of seed information as follows:

$$b_j^i \leftarrow SS.\text{Share}\left(t, n, b_j\right) \tag{16}$$

$$\text{seed}_{i,j}^{sk} \leftarrow \text{SS.Share}\left(t, n, \text{ seed }_i^{sk}\right) \tag{17}$$

$$\text{seed}_{i,j}^{e} \leftarrow \text{SS.Share}\left(t, n, \text{ seed }_{i}^{e}\right) \tag{18}$$

Data owners running the secret sharing P_i, use the symmetric key $key_{i,j}^{c}$ to encrypt the secret shares, and transmit them through the central server to other participants. If the number of online participants, $|U_2| < t$, this round of federated learning execution is terminated. Finally, the public vector parameters generated by the secret sharing algorithm are disclosed.

Model aggregation stage
During the model aggregation phase, the server receives encrypted gradient information y_i^{k+1} from online data owners and performs gradient aggregation:

$$y^{k+1} = \sum_{i \in U_3} y_i^{k+1}, t \leq |U_3| \leq n \tag{19}$$

If the number of online users $|U_3|$ is less than $t(|U_3| < t)$, the protocol is terminated. For other users $P_i \notin U_3$ who have not dropped out and are participating in the aggregation, $i \neq j$, P_j, online participants use secret shares related to b_i to compute secret information. Otherwise, secret shares related to $key_{i,j}^{s}$ are selected for computation. After computing the shares using the public parameters, the training uploads the calculated information to the server.

The server collects all received secret information. If the number of shares is less than *t*, the protocol is terminated. In the PQSF scheme, for users $P_i \notin U_3$ who have dropped out, the server runs the secret reconstruction algorithm *SS.Recon*() to restore the symmetric key. And the secret shares of the remaining participants exceeding the threshold are collected to reconstruct the masks of the offline participants. In addition, under the double mask mechanism, the model gradient of each participant is encrypted by both the key-based mask and the random mask. Even if some participants are offline, the remaining gradients can be securely aggregated to prevent personal data leakage. Before the training starts, the participants distribute the secret shares of the random seeds used for masking through the threshold secret sharing scheme. Once a participant goes offline, the remaining participants can use these secret shares to reconstruct the lost random mask, allowing the server to accurately aggregate the gradients. This process effectively ensures the accuracy and security of gradient aggregation, so that it can calmly cope with the challenges brought by the disconnection of participants.For data owners $P_j \notin U_3$ participating in the aggregation calculation, the central server restores the mask b_j and calculates the mask $PRG(b_j)$ using the secret reconstruction algorithm *SS.Recon*(). Finally, the server uses these masks to calculate the aggregated result and update the new global model w_i^{k+1}.

Subsequent training stage
In the subsequent rounds of training, participants negotiate new symmetric keys and locally generate random masks. Training participants run the secret renewal algorithm *SS.Renew*() to obtain new public parameters for the next round of secrets and make them public. During this process, the Improved-Pilaram multi-stage secret sharing scheme allows participants to use the same secret shares in multiple stages without frequently updating these shares. In this way, in each training round, participants do not need to redistribute secret shares, but only need to use new public parameters for secret reconstruction.

In the subsequent secret reconstruction phase, the data holders participating in the reconstruction use the original secret share and new public parameters to calculate the encrypted information without the need to frequently update or re-transmit these public parameters. After the server receives a sufficient amount of information, it runs the secret reconstruction algorithm to obtain a new mask and calculates the final aggregated gradient information. With the reuse of secret shares and the sharing of public parameters, the communication overhead during each round of training is significantly reduced. The federated learning algorithm repeats these steps until the model converges or a specified number of training epochs is reached to terminate.

Security analysis

Correctness
Theorem 5 *The execution process of the proposed PQSF scheme ensures that the server can always compute and output the correct gradient aggregation results, given the number of online training participants participating in model aggregation, denoted as* $|U| \leq t$.

Proof Through the NewHope key exchange scheme, participants P_i and P_j can negotiate and obtain a shared key $key_{i,j}^{s}$=$key_{j,i}^{s}$. We first assume that during the training process, all individual training participants have not dropped out midway and have encrypted the gradients according to the protocol and participated in secret reconstruction. From the gradient encryption algorithm, we can derive the correctness of Eq. (20).

$$\begin{aligned} \sum_{i=1}^{n} y_i^{k+1} &= \sum_{i=1}^{n} \left(w_i^{k+1} + PRG\left(b_i\right) + \sum_{j \in U, i<j} PRG\left(key_{i,j}^{s}\right) - \sum_{j \in U, i>j} PRG\left(key_{i,j}^{s}\right) \right) \\ &= \sum_{i=1}^{n} w_i^{k+1} + \sum_{i=1}^{n} PRG\left(b_i\right) \end{aligned} \tag{20}$$

During the gradient aggregation phase, participants upload secret shares regarding the mask b_i, and the central server runs a secret reconstruction algorithm to obtain $b_i, i = 1, 2, \ldots, n$ and ultimately eliminate the mask, resulting in the aggregated model gradient. Next, we consider the scenario of participants dropping out during the training process.

$$\begin{aligned}\sum_{i=1}^{|U|} y_i^{k+1} &= \sum_{i=1}^{n}\left(w_i^{k+1} + PRG\left(b_i\right) + \sum_{j\in U, i<j} PRG\left(key_{i,j}^{s}\right) - \sum_{j\in U, i>j} PRG\left(key_{i,j}^{s}\right)\right) \\ &= \sum_{i=1}^{n} w_i^{k+1} + \sum_{i=1}^{n} PRG\left(b_i\right) \pm \sum PRG\left(key^{s}\right)\end{aligned} \tag{21}$$

When there are participants dropping out during the training process, the server still computes the correct gradient aggregation result. Firstly, assuming that the training participants $P_l, l \notin U$ drop out midway, participating only $P_i, i \in U$ in the final model aggregation and secret reconstruction, the server can verify the correctness of Eq. (21) through aggregating encrypted gradients.

Here, $PRG(key^s)$ denotes the uncompensated mask due to participant P_l dropout. In the secret reconstruction phase, training participants make decisions: for users in the online set *U*, they calculate and upload the secret information corresponding to the mask b_i using public parameters. For dropout users not belonging to the set *U*, they upload secret information related to P_l symmetric keys. The central server can ultimately eliminate the shares and perform model aggregation through the reconstruction algorithm *SS.Recon*().

In conclusion, as long as the number of online participants involved in model aggregation and secret reconstruction $|U| \leq t$, the correct model aggregation result can be successfully computed, thereby proving Theorem 5.

Privacy protection

Theorem 6 *The proposed PQSF scheme ensures the security of participant models. In other words, honest but curious participants or the central server cannot compromise the security of the local data sets of participants through inference attacks.*

Proof In the assumed semi-honest model, training participants and the server correctly encrypt and aggregate gradients according to the protocol. For the gradients of participant models, using masked encryption ensures that the original and encrypted gradients are indistinguishable. Suppose adversary A steals ciphertext information during the participant's model gradient upload process. However, due to the security of the NewHope key exchange scheme in the dual-mask scheme and the security of *Improved-Pilaram*, Theorem 3 and Theorem 4 can guarantee the privacy of original gradients in the *PQSF*.

Furthermore, during the aggregation process of the model by the central server, the security of participant local gradients is also ensured. This is because the proposed scheme in this paper not only uses symmetric key generation for masks that can cancel each other out but also encrypts gradients using dual masks based on random numbers b_i. Even if honest but curious participants have received gradient vectors from participants and aggregated them to eliminate all symmetric key masks, the privacy of local data can still be ensured through random number masking. Honest but curious servers cannot compromise the security of participant local data sets through inference attacks on gradients, thus proving Theorem 6.

Simulation experiment and analysis

In this section, we conducted simulation experiments on both the proposed *Improved-Pilaram* and the *PQSF* to demonstrate the feasibility of our approach. In addition, this section compares the proposed scheme with existing schemes and demonstrates the advantages of the proposed scheme in terms of computational overhead. The simulation experiments in this section were conducted on terminals running 64-bit Windows 10 systems. We simulated participants and server on a desktop computer configured with an Intel i7-9700 CPU @3.7 GHz and 64GB of installed memory.

Improved-Pilaram

This section evaluates the performance of the proposed *Improved-Pilaram*, analyzing the secret sharing, secret reconstruction, and secret updating phases of the protocol. The experiment examines the relationship between computational overhead and threshold ratio *t*/*n* under different numbers of participants, and compares it with the post-quantum secure scheme Mus. Specifically, this section simulates computational overhead at different stages with 100, 200, 300, 400, and 500 participants, with thresholds *t*/*n* ranging from 0.5 to 0.9.

Time overhead in the secret sharing stage

The time overhead of the proposed secret sharing algorithm at different thresholds is shown in Fig. 4a of this section. From the experiments, it can be observed that the computational overhead of participants increases linearly with the increase in the threshold *t*/*n*. Additionally, the increase in the number of participants also has a similar effect on the scheme. This is because the size of shares and the public matrix proposed in this section are both correlated with the threshold ratio. When there are 300 participants, the sharing algorithm takes 1403.4 ms to run at a threshold *t*/*n* of 0.5, and 1813.7 ms to run at a threshold *t*/*n* of 0.7, and 2261.5 ms to run at a threshold

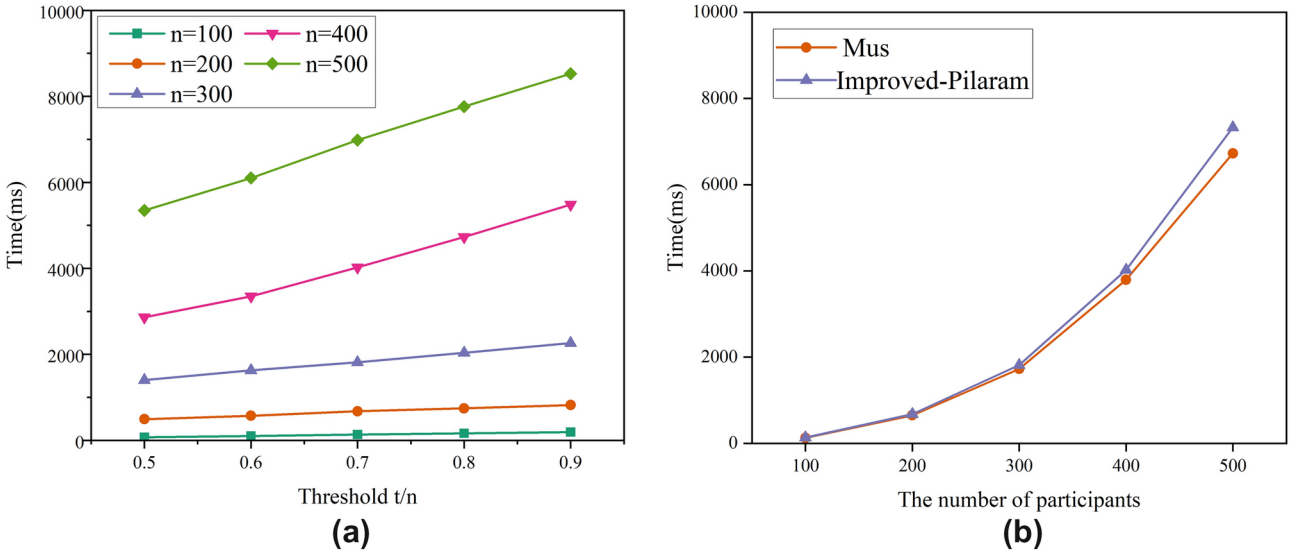

Figure 4. Secret sharing phase.

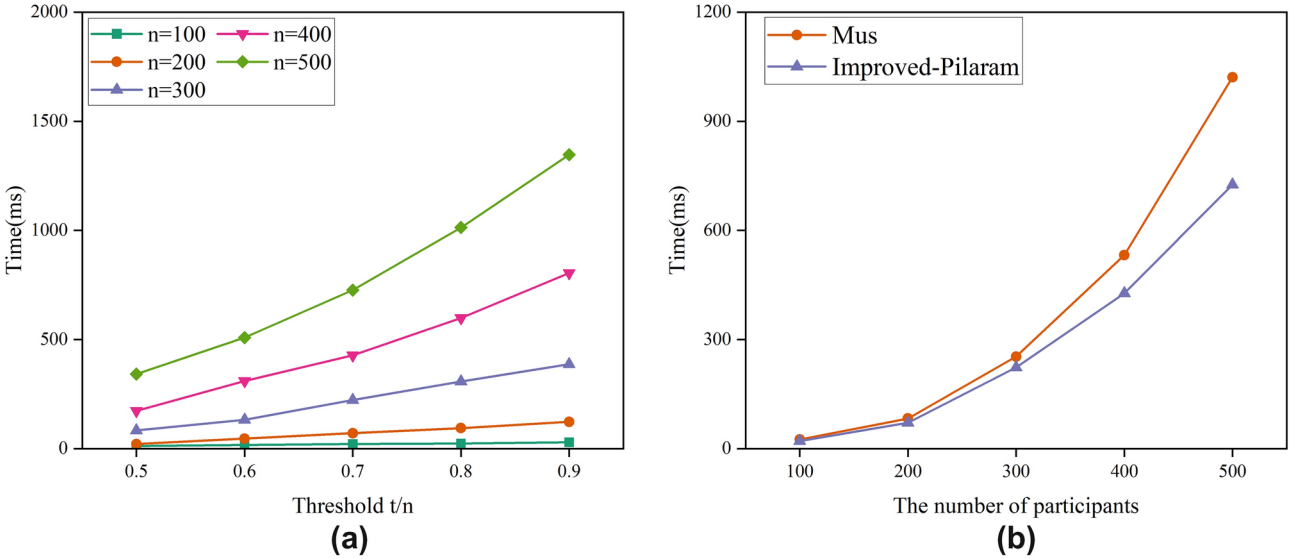

Figure 5. Secret reconstruction phase.

t/n of 0.9. When the threshold t/n is fixed at 0.7, the algorithm requires 137.2 ms to run with 100 participants, 1813.5 ms with 300 participants, and 6980.1 ms with 500 participants.

Figure 4b illustrates the comparative experiments between the approach proposed in this paper and Mus. The modified approach incurs some additional computational overhead in the secret sharing phase. This is because the *Improved-Pilaram* proposed in this paper adds an additional random vector during the construction process to ensure the security of secrets during multiple rounds of reconstruction. When the number of participants is 300 and the threshold t/n is fixed at 0.7, the time taken for the secret sharing phase in the Mus scheme is 1725.2 ms, while the proposed approach in this paper takes 1813.5 ms. The additional computational overhead is within an acceptable range.

Time overhead in the secret reconstruction stage

During the secret reconstruction phase, training participants locally compute secret information using shares and public vectors, and the secret reconstruction operation is completed by the initiator. As shown in Fig. 5a, the computational overhead in the reconstruction phase increases linearly with the number of participants. When there are 300 participants, the reconstruction algorithm requires 83.7 ms to run at a threshold t/n of 0.5, 223.4 ms at a threshold t/n of 0.7, and 387.9 ms at a threshold t/n of 0.9. When the threshold t/n is set to 0.7, the

reconstruction algorithm consumes 21 ms with 100 participants, 308 ms with 300 participants, and 726 ms with 500 participants.

In the Mus scheme, secret shares from participants are uploaded, and the secret reconstructor completes share calculation and reconstruction based on public parameters. However, in the approach proposed in this paper, participants locally perform combined operations on shares and public parameters to obtain secret information before uploading. This approach ensures the privacy of shares while also reducing the computational overhead for the secret reconstructor. As seen in Fig. 5b, with an increasing number of participants, the proposed approach exhibits lower computational overhead in the reconstruction phase compared to Mus. In real-world scenarios, participant local computations are distributed and synchronized, and the aggregator only needs to collect sufficient secret information from participants for combined computation. This significantly reduces the overall system's computational overhead and time, while ensuring the security of secret shares is not compromised by aggregation.

Time overhead in the secret update stage

In the improved *Improved-Pilaram* of this paper, during the update phase, the original secret shares are used to compute the updated public vectors. In this process, the Dealer saves computational overhead by not generating multiple random vectors, and participants save overhead by not updating secret shares. As shown in Fig. 6a, when the number of participants is 300 and the threshold ratio t/n is 0.5, 0.7, and 0.9, the secret update algorithm requires 731.6, 995.4, and 1349.6 ms, respectively.

As shown in Fig. 6b, the proposed approach in this paper exhibits lower computational overhead during the update phase compared to Mus. The primary advantage in computational overhead lies in the fact that secret shares and certain vector parameters do not need to be modified, as the privacy of shares can be maintained over the long term in multi-secret sharing processes. However, in environments where all users share and update secrets, the proposed approach in this paper has smaller computational overhead, which is crucial for federated learning environments.

In summary, simulation experiments indicate that the *Improved-Pilaram* proposed in this section has lower computational overhead overall compared to Mus. Another significant advantage of this scheme is that secret shares do not change with secret updates, which is further amplified in scenarios with multiple participants and multiple secrets. Additionally, the scheme provides post-quantum security, which traditional secret sharing algorithms do not possess.

PQSF

In this section, we conducted simulation experiments on the proposed *PQSF* model aggregation scheme. As the approach used in the scheme is based on dual masking, the final model accuracy is not compromised by the implementation of privacy protection. Therefore, this section does not analyze the accuracy of the model. We conduct experiments with convolutional neural networks (CNN) on the MNIST data set. The hidden layers of the CNN consists of two 5×5 convolution layers followed by max polling layers, and two fully connected layers.

We simulated and compared the time overhead of the proposed privacy-preserving federated learning method in a single training round. Due to the construction of multi-stage secret sharing, we separately tested the overhead for participants in the initial stage and subsequent training stages, and compared it with the Laf scheme. In this simulation experiment, we set the threshold t/n for secret sharing to 0.7 and the proportion of participant dropouts to 70%. As shown in Fig. 7, although the secure aggregation scheme proposed in this paper has a large computational overhead in the initial rounds, the time overhead in subsequent training stages

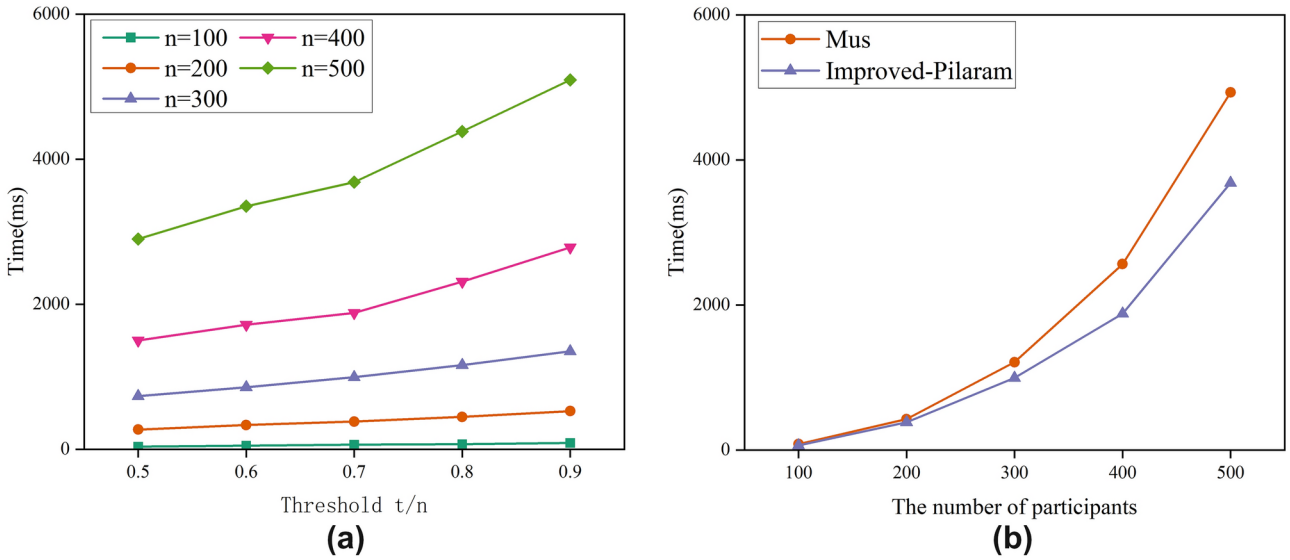

Figure 6. Secret update phase.

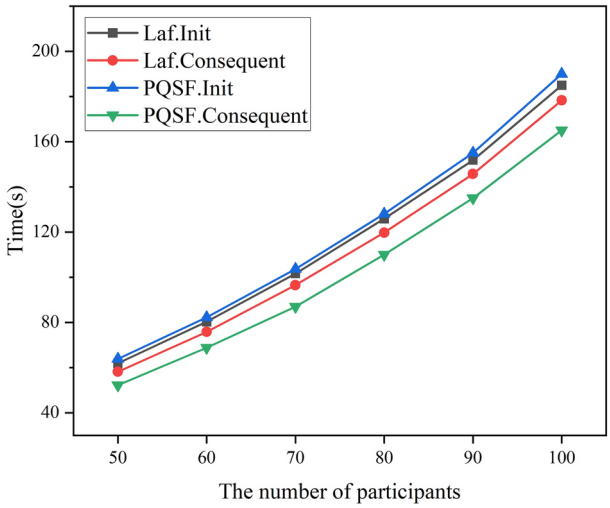

Figure 7. Comparison of single-round time overhead under different numbers of participants.

Scheme	Time (ms)		Public key size (bits)	Private key size (bits)	Communication overhead (bits)	Post-quantum security
	Alice	Bob				
NewHope	0.326	0.375	14592	14336	30976	Support
DH	0.914	0.915	256	256	512	Unsupported

Table 1. Comparison of the overhead of the newhope key agreement scheme.

is significantly reduced. This is due to the fact that the protocol requires more vector computations during the initial secret sharing phase, while these vectors and secret shares do not need to be regenerated during the secret update and subsequent phases. Therefore, our method exhibits lower computational and communication burden throughout the federated learning process.

In addition, the federated learning scheme proposed in this paper requires key agreement among participants. Therefore, we compared the computational overhead and the sizes of public and private keys between the NewHope key agreement protocol and traditional Diffie–Hellman (DH) key agreement protocol. As shown in Table 1, although there are discrepancies in communication volume and key size compared to traditional algorithms, the NewHope key agreement protocol used in this paper exhibits strong computational efficiency advantages, even surpassing traditional key agreement schemes. In conclusion, the key agreement scheme used in this paper, while ensuring post-quantum security, maintains acceptable sizes of public and private keys and efficient computational overhead.

This section conducted simulation experiments to test the computational overhead of the multi-stage *Improved-Pilaram* at different stages. Additionally, the computational overhead of the federated learning protocol based on this technique and the Key Agreement protocol were also tested. Through comparative experiments, the results indicate that the proposed approach in this section has smaller time overhead in federated learning. Furthermore, this approach does not require frequent updates of local secret shares during execution, making it more practical in real-world scenarios, and it can provide post-quantum security for model safety.

Conclusion

This paper presents a post-quantum secure federated learning secure aggregation scheme utilizing lattice-based secret sharing techniques. It aims to reduce communication overhead among participants while protecting the privacy of model parameters. This paper introduces a multi-stage *Improved-Pilaram* that enables participants to reconstruct different secrets using local shares and public parameters. In the proposed *PQSF*, this scheme encrypts model parameters using lattice-based key agreement scheme and dual masking mechanisms to protect them. It also implements mask reconstruction elimination based on the improved secret sharing scheme and ensures robustness against dropout participants. On one hand, participants in federated learning do not need to frequently update local secret shares, reducing computational overhead. On the other hand, participants no longer need to transmit shares to each other after the initial sharing round, reducing communication overhead. The security proofs and simulation experiments demonstrate that the proposed approach effectively reduces computational overhead while ensuring post-quantum security.

Data availability

The data used to support the findings of this study are available from the corresponding author on reasonable request.

Received: 14 April 2024; Accepted: 24 September 2024
Published online: 09 October 2024

References

1. McMahan, B., Moore, E., Ramage, D., Hampson, S. & y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, 1273–1282 (PMLR, 2017).
2. Li, L., Fan, Y., Tse, M. & Lin, K.-Y. A review of applications in federated learning. *Comput. Ind. Eng.* **149**, 106854 (2020).
3. Fu, A. et al. Vfl: A verifiable federated learning with privacy-preserving for big data in industrial iot. *IEEE Trans. Ind. Inform.* **18**, 3316–3326 (2020).
4. Fang, G. et al. Distributed medical data storage mechanism based on proof of retrievability and vector commitment for metaverse services. *IEEE J. Biomed. Health Inform.* (2023).
5. Liu, Y. et al. Privacy protection techniques in federated learning. *J. Softw.* **33**, 1057–1092 (2021).
6. Li, Q. et al. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Trans. Knowl. Data Eng.* **35**, 3347–3366 (2021).
7. Gong, M., Xie, Y., Pan, K., Feng, K. & Qin, A. K. A survey on differentially private machine learning. *IEEE Comput. Intell. Mag.* **15**, 49–64 (2020).
8. Ma, J., Naas, S.-A., Sigg, S. & Lyu, X. Privacy-preserving federated learning based on multi-key homomorphic encryption. *Int. J. Intell. Syst.* **37**, 5880–5901 (2022).
9. Zhang, C., Ekanut, S., Zhen, L. & Li, Z. Augmented multi-party computation against gradient leakage in federated learning. *IEEE Trans. Big Data* (2022).
10. Ren, Y. et al. Multiple cloud storage mechanism based on blockchain in smart homes. *Future Gen. Comput. Syst.* **115**, 304–313 (2021).
11. Xu, G., Li, H., Liu, S., Yang, K. & Lin, X. Verifynet: Secure and verifiable federated learning. *IEEE Trans. Inf. Forensics Secur.* **15**, 911–926 (2019).
12. Yin, L., Feng, J., Xun, H., Sun, Z. & Cheng, X. A privacy-preserving federated learning for multiparty data sharing in social iots. *IEEE Trans. Netw. Sci. Eng.* **8**, 2706–2718 (2021).
13. Bernstein, D. J. & Lange, T. Post-quantum cryptography. *Nature* **549**, 188–194 (2017).
14. Ren, Y., Leng, Y., Cheng, Y. & Wang, J. Secure data storage based on blockchain and coding in edge computing. *Math. Biosci. Eng.* **16**, 1874–1892 (2019).
15. Pilaram, H. & Eghlidos, T. An efficient lattice based multi-stage secret sharing scheme. *IEEE Trans. Depend. Secure Comput.* **14**, 2–8 (2015).
16. Yang, Z., He, D., Qu, L. & Xu, J. On the security of a lattice-based multi-stage secret sharing scheme. *IEEE Trans. Depend. Secure Comput.* (2022).
17. Xu, P., Hu, M., Wang, W. & Jin, H. Laf: Lattice-based and communication-efficient federated learning. *IEEE Trans. Inf. Forensics Secur.* **17**, 2483–2496 (2022).
18. McMahan, B., Moore, E., Ramage, D. et al. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 1273–1282 (2017).
19. Sun, L., Li, C., Ren, Y. & Zhang, Y. A multitask dynamic graph attention autoencoder for imbalanced multilabel time series classification. *IEEE Trans. Neural Netw. Learn. Syst.* (2024).
20. Song, M. et al. Analyzing user-level privacy attack against federated learning. *IEEE J. Sel. Areas Commun.* **38**, 2430–2444 (2020).
21. Xiong, X. et al. A comprehensive review of privacy protection and security defense in federated learning. *J. Comput. Sci. Technol.* **46**, 1019–1044 (2023).
22. Zhao, Y. et al. Local differential privacy-based federated learning for internet of things. *IEEE Internet Things J.* **8**, 8836–8853 (2020).
23. Jia, B. et al. Blockchain-enabled federated learning data protection aggregation scheme with differential privacy and homomorphic encryption in iiot. *IEEE Trans. Ind. Inform.* **18**, 4049–4058 (2021).
24. Bonawitz, K., Ivanov, V., Kreuter, B. et al. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 1175–1191 (ACM, 2017).
25. Duan, J., Zhou, J. & Li, Y. Privacy-preserving distributed deep learning based on secret sharing. *Inf. Sci.* **527**, 108–127 (2020).
26. Shamir, A. How to share a secret. *Commun. ACM* **22**, 612–613 (1979).
27. Blakley, G. R. Safeguarding cryptographic keys. In *Proceedings of the AFIPS*, 313–318 (IEEE Computer Society, 1979).
28. Zhang, Y. et al. Protect: Efficient password-based threshold single-sign-on authentication for mobile users against perpetual leakage. *IEEE Trans. Mob. Comput.* **20**, 2297–2312 (2020).
29. Sun, L., Wang, Y., Ren, Y. & Xia, F. Path signature-based xai-enabled network time series classification. *Sci. China Inf. Sci.* **67**, 170305:1-170305:16 (2024).
30. Kandar, S. & Dhara, B. C. A verifiable secret sharing scheme with combiner verification and cheater identification. *J. Inf. Secur. Appl.* **51**, 102430 (2020).
31. Chandramouli, A., Choudhury, A. & Patra, A. A survey on perfectly secure verifiable secret-sharing. *ACM Comput. Surv. (CSUR)* **54**, 1–36 (2022).
32. Shor, P. W. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 124–134 (IEEE, 1994).
33. Chao, W. et al. Progress in cryptographic attacks via quantum computing. *J. Comput. Sci. Technol.* **43**, 1691–1707 (2020).
34. Rajabi, B. & Eslami, Z. A verifiable threshold secret sharing scheme based on lattices. *Inf. Sci.* **501**, 655–661 (2019).
35. Gentry, C., Halevi, S. & Lyubashevsky, V. Practical non-interactive publicly verifiable secret sharing with thousands of parties. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 458–487 (Springer, 2022).
36. Ravi, P., Howe, J., Chattopadhyay, A. & Bhasin, S. Lattice-based key-sharing schemes: A survey. *ACM Comput. Surv. (CSUR)* **54**, 1–39 (2021).
37. Binwu, X., Jiang, Z. & Yi, D. Introduction to lattice-based public key encryption and key encapsulation mechanisms among nist post-quantum cryptography standard candidates. *J. Cryptol. Res.* **10**, 20–45 (2023).

Acknowledgements

This work is supported by the Key Project of Jiangsu Collaborative Innovation Center of Chinese Medicinal Resources Industrialization (ZDXM-2023-18), the General Project of Philosophy and Social Science Research in Colleges and Universities in Jiangsu Province (2021SJA0337). This work is also supported by the Natural Science Foundation of China (No. 62072249).

Author contributions

Xia Zhang: Conceptualization, Methodology, Software, Writing-original draft. Haitao Deng: Investigation, Data curation, Writing—review and editing. Rui Wu: Conceptualization, Resources, Writing-review and editing.

Jingjing Ren: Software, Writing—original draft, Data curation. Yongjun Ren: Funding acquisition, Supervision, Project administration.

Declarations

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to Y.R.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.