electronics

Article

Non-Interactive and Secure Data Aggregation Scheme for Internet of Things

Yanxia Fu [1], Yanli Ren [1,*], Guorui Feng [1], Xinpeng Zhang [1] and Chuan Qin [2]

[1] School of Communication and Information Engineering, Shanghai University, Shanghai 200444 , China; yanxia_fu@shu.edu.cn (Y.F.); grfeng@shu.edu.cn (G.F.); xzhang@shu.edu.cn (X.Z.)

[2] School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China; qin@usst.edu.cn

* Correspondence: renyanli@shu.edu.cn

Abstract: The popularity of mobile devices in Internet of Things has brought great convenience to the lives of the people. Massive data generated in the IoT are outsourced and stored on cloud platforms so that data aggregation and analysis can be performed on the massive data. However, these data often contain sensitive information of mobile devices, so effective protection of mobile user privacy is the primary condition for further development of IoT. Most of the current data aggregation schemes require a lot of interactions between users, and thus this paper designs a non-interactive secure multidimensional data aggregation scheme. This scheme adopts an additive secret sharing technique to mask the shared data and send it to two non-colluding servers, and then the servers aggregate the ciphertext respectively. Different from the existing schemes, our proposed scheme achieves non-interaction between users, and the aggregation result is kept confidential to the server and supports mobile users offline. Finally, we perform an experimental evaluation which proves the effectiveness of our scheme.

Keywords: privacy-preserving; secret sharing; verification; robustness; Internet of Things

check for updates

Citation: Fu, Y.; Ren, Y.; Feng, G.; Zhang, X.; Qin, C. Non-Interactive and Secure Data Aggregation Scheme for Internet of Things Non-Interactive and Secure Data Aggregation Scheme for Internet of Things. *Electronics* **2021**, *10*, 2464. https://doi.org/10.3390/electronics10202464

Academic Editors: Cheng-Chi Lee, Sedat Akleylek and Paulo Ferreira

Received: 10 August 2021
Accepted: 5 October 2021
Published: 11 October 2021

1. Introduction

With the rapid development of mobile devices and cloud computing, the IoT has brought great convenience to the lives of people. According to the mobile economy 2020 released by the Global System for Mobile Communications Assembly, the total number of global IoT connections reached 12 billion in 2019, and they will reach 24.6 billion in 2025 [1]. These devices will generate data, which means that the IoT can have a deeper understanding of user behavior, so as to provide users with more convenient and faster services.

The large amount of data generated by the IoT covers various fields, such as medical life services, economy and education, which have promoted the development of human society. However, in the face of these massive amounts of data, the information in their collection, processing and transmission process may be maliciously attacked, which may lead to privacy information leakage. For individuals, smart wearable devices record information about our daily life, including our health status, location, call records, etc. However, the transmission process between the device and the mobile phone is not secure, and once the private information is intercepted by malicious attackers, the consequences are beyond imaginable. For the country, once the confidential national information is stolen by illegal elements, the security is doomed to face huge losses, so the importance is self-evident.

Aggregation technology is a common data processing method in the IoT. Smart devices generate extensive data which are collected and aggregated by cloud servers and applied in learning algorithms such as machine learning and artificial intelligence. The data are analyzed and predicted so as to improve the user experience of using smart devices [2,3], and cloud servers provide basic services for IoT and improve the computing efficiency of smart devices. For example, if you want to know the electricity situation of users in an area, the cloud server will aggregate the electricity information of these users. After that, the electricity service provider could analyze the total electricity consumption, so as to reasonably allocate the electricity resources in this area. However, they are also facing the issue of data privacy leakage, so data security and user privacy have also received great attention [4–7].

In the existing secure data aggregation research, most of the schemes are able to protect the private data of users well, but they use complex encryption techniques, which leads to high computational overhead and communication costs for users. In addition, due to the complexity of the real environment, mobile devices may drop out at any time during the process of performing aggregation tasks. However, many schemes do not take this problem into consideration.

In order to work out the problem of the huge communication cost, and support users offline and realize the verification of the aggregation results, this paper proposes a non-interactive and secure data aggregation algorithm, which means that the mobile users do not interact with each other. Compared with the previous ones, our algorithm not only reduces the computational complexity of local users from $O(n^2 + mn)$ to $O(m)$, but also reduces the communication cost from $2(3n + 1)m|p|$ to $3m|p|$. In addition, this scheme supports users offline and can effectively verify the aggregation results. Our contributions are as follows:

1. We design a non-interactive and secure data aggregation scheme that has low communication overheads, which is robust to the exiting users and supports mobile users offline. This scheme uses additive secret sharing to share the original data in two parts, and then masks two shared values with a random number. Finally, the ciphertext is sent to two non-colluding cloud servers separately. Compared with the previous aggregation ones, our scheme reduces the computation and communication costs of users.
2. In most of the previous schemes, the server can obtain the final aggregation results, and it is possible for the cloud to misuse the aggregation results for malicious analysis and speculation. However, in our scheme, the two cloud servers perform data aggregation with the ciphertext of the aggregation result, and therefore the true aggregation result is well-protected from the servers.
3. This paper designs a set of algorithms so that the final aggregation result can be efficiently verified. Anyone can check the correctness of the aggregation results with a probability of 1, and it is impossible for an incorrect aggregation result to be successfully verified.

The rest of this paper is organized as follows. In Sections 2 and 3, we first present the related work and preliminaries. The system framework and the non-interactive and secure data aggregation scheme are introduced in Section 4. Privacy and limitations of the paper are discussed in Section 5. The experimental results are presented in Section 6. Finally, we outline the conclusions of the paper in Section 7.

2. Related Work

In the past few decades, scholars have put forward many secure data aggregation schemes from different applications, mainly on account of secure multi-party computation, differential privacy and homomorphic encryption. Due to the emergence of federated learning, the problem of data islands has been well-resolved. Participants calculate the training gradient locally, then use encryption technology to mask the local gradient before sending the ciphertext to the server. After that, the server aggregates these ciphertexts securely, and then returns the aggregation results to the participants. This process does not require participants to directly share local data, thereby reducing many privacy leaks [8–11].

In the field of machine learning, Bonawitz et al. [12] provided a double-masked secure aggregation scheme by using key agreement and secret sharing techniques, which not only ensures user input privacy but is also robust to dropped users. However, there are many shortcomings in this scheme, such as the need for continuous interaction between users, the lack of support for offline users and the inability to verify aggregation results. On the basis of the above scheme, Xu et al. [13] proposed a similar scheme based on federated learning which verifies the aggregation results and ensures the correctness of aggregation. Yang et al. [14] achieved an aggregation scheme with paillier homomorphic encryption, which ensures the privacy of user input at the same time. Mandal et al. [15] proposed the practical privacy-preserving federated regressions scheme, which can train regression models in a federated learning environment. At the same time, it ensures data and model privacy. Based on proxy encryption and aggregate signature techniques, Xu et al. [16] designed a multi-party secure learning framework. The main feature of this scheme is to use proxy encryption to realize key transformation, so that the server does not obtain the final aggregation result and only the user can utilize the corresponding private key to decrypt the aggregation result.

In the application of the Internet of Things, secure data aggregation is more widely used. Based on homomorphic encryption technology, Al-Zubaidie et al. [17] used a secure and lightweight signature algorithm to prevent user information leakage. Edemacu et al. [18] proposed a multi-party privacy-preserving logistic regression framework and filtered out low-quality data for data contributors. Yang et al. [19] designed an encryption scheme on account of elliptic curve encryption, which reduces the computation and communication costs. However, it does not propose a verification algorithm to verify the aggregation results. Li et al. [20] put forward an aggregation scheme by using key agreement and a verification algorithm. Although the scheme proposes a verification algorithm for aggregating results, it does not support user drop out, and therefore, once the user drops out, this aggregation protocol will aggregate incorrect results. Based on Bonawitz's scheme, Jiang et al. [21] added the functions of location privacy and multi-dimensional data analysis for mobile users, but the communication cost is still very high and most mobile users must be online.

3. Preliminaries

This section will introduce some basic concepts, including secure multi-party computation, additive secret sharing, pseudo-random generator, bilinear map and verifiable computation.

3.1. Secure Multi-Party Computation

Yao proposed a theoretical framework for secure multi-party computation, which resolves the problem of collaborative computation among a set of mutually distrustful parties in the context of protecting private information without a trusted third party [22].

Afterwards, several papers [23–26] applied this theory. Secure multi-party computation means the collaborative computation of an agreed function by multiple participants with an untrusted third party. It is also guaranteed that each party only obtains its own calculation results and cannot infer the input and output data of any other party from the interaction data during the calculation. The mathematical formula for multi-party security computation is as follows:

$$f(a_1, a_2, ..., a_n) = (b_1, b_2, ..., b_n) \tag{1}$$

where $a_1, a_2, ..., a_n$ are the inputs of each participant, $b_1, b_2, ..., b_n$ are the corresponding outputs and f is the computation function agreed by each participant. Each participant can only see their own inputs and outputs in the whole multi-party computation protocol, and cannot know the inputs and outputs of other participants.

3.2. Additive Secret Sharing

Inspired by former works [27,28], this paper introduces additive secret sharing to split the data into two parts. We assume a mobile user O_i shares the secret u (define $< u >$ as the shared value of u) to server A and server B, then S_A and S_B obtain $< u >_A$ and $< u >_B$, where $< u >_A + < u >_B = u$ mod p. It is worth noting that S_A and S_B are non-colluding, so $< u >_A$ ($< u >_B$) is only known to S_A (S_B). In order to reconstruct the value of u later, two parties need to send the secret shared value together to obtain u. In our scheme, the data are first split into two secret shared values, which not only protects the original data, but also achieves non-interactivity between the users.

3.3. Pseudo-Random Generator

A pseudo-random number generator [29,30] is used to reduce our computational cost in our scheme. A random number seed is used to generate irrelevant random numbers and we use PRG (seed) to represent the output random numbers. Pseudo-random generator: $\{0,1\}^n \rightarrow \{0,1\}^l$ is a deterministic that receives an n-bit binary string as input and outputs an l-bit binary string, where l is much larger than n. The security of a cryptographically strong PRG is defined by the fact that its output is indistinguishable from the bit stream of a true random number.

3.4. Bilinear Map

Let G_1, G_2 and G_τ be cyclic groups of order p, where p is a large prime, and g_1 and g_2 are generators of G_1, G_2, respectively. e: $G_1 \times G_2 \rightarrow G_\tau$ is a bilinear map, which satisfies the following conditions [31]:

1. Bilinearity: For any $a, b \in Z_p^*$ and $g \in G_1, h \in G_2$, we have $e(g^a, h^b) = e(g,h)^{ab}$.
2. Computability: There is an efficient algorithm to compute $e(g,h)$ for $g \in G_1, h \in G_2$.
3. Non-degeneracy: There exists $g \in G_1, h \in G_2$, such that $e(g,h) \neq 1$.

BDHE Problem: The Bilinear Diffie-Hellman Exponent (BDHE) problem is described as follows: given $g, g^a, g^b \in G_1, h \in G_2$, the output is $e(g,h)^{ab}$.

BDHE assumption: Given $g, g^a, g^b \in G_1, h \in G_2$, for any $a, b \in F_p^*$, if the probability to solve the BDHE problem is negligible, it means that the BDHE assumption holds in G_τ.

3.5. Verifiable Computation

The verification algorithm is usually used to check the results which are returned by the server. The main purpose of the References [32–35] is to study verifiable algorithms. Since cloud servers are untrustworthy, Jia et al. [36] and Zhang et al. [37] proposed public verification algorithms supporting large-scale matrix multiplication. In our scheme, the verification algorithm allows the verifier to verify the final result. The algorithm contains the following four sub-algorithms:

1. KeyGen $(f, \lambda) \rightarrow (pk, sk)$: The inputs to the key generation algorithm are the secure parameter λ and function f, and then the outputs are pk and sk.
2. ProbGen $(sk, x) \rightarrow (\sigma_x, \tau_x)$: The inputs to the problem generation algorithm are sk and x, and the outputs are σ_x, τ_x, where σ_x is a public value and τ_x is a private value kept by the user.
3. Compute $(pk, \sigma_x) \rightarrow \sigma_y$: The algorithm takes pk and σ_x as inputs, and the cloud server computes the output value σ_y.
4. Verify $(sk, \tau_x, \sigma_y) \rightarrow y \cup \perp$: The user uses sk and τ_x to verify whether σ_y is correct; if the verification passes, it will be accepted, otherwise, it will be rejected.

4. System Framework and Non-Interactive and Secure Data Aggregation Scheme

In this section, the system framework and the non-interactive and secure data aggregation scheme are presented in detail.

4.1. System Framework

Here, we introduce the system model, threat model and design goals of this paper.

4.1.1. System Model

The system model consists of four roles: mobile users, two non-colluding aggregation servers, control center and public verifier.

Mobile users: In the system, n mobile users are set up (defined as $O = \{O_1, O_2, ..., O_i, ...O_n\}$), and each mobile user holds a local private vector and uses a random number to protect the private vector. At the same time, it also generates a verification key.

Cloud servers: This system consists of two non-colluding servers, S_A and S_B, mainly to realize non-interactivity between users and keep the aggregation results confidential to the cloud server. They will aggregate these vectors and also generate the corresponding proofs when they receive masking vectors from n mobile users. Then, the aggregation results and proofs are sent to the control center.

Control center: The two aggregation results are added up to obtain the final result after the control center receives the aggregation results from S_A and S_B, so that the data of these users can be analyzed by the control center.

Public verifier: The verifier usually refers to trusted institutions that can check the final result based on the proofs and verification keys.

As shown in Figure 1, each mobile user first sends the local ciphertexts $< c_i >_A$ and $< c_i >_B$ to S_A and S_B through a secure channel, respectively. Then, the server aggregates the ciphertexts of all users and generates an aggregation value and verification key. Next, the control center computes the final aggregation result. Finally, the public verifier checks the final aggregation result according to the verification keys of the mobile user and the server. In Table 1, we list the main symbols used in this paper, respectively.

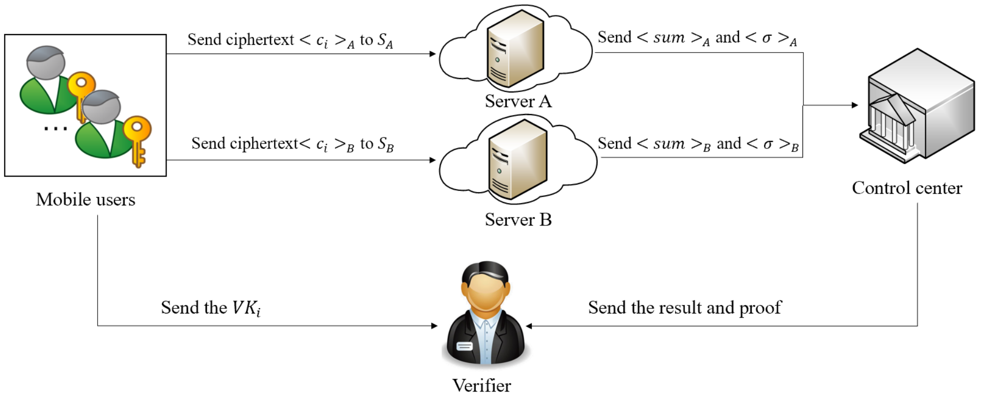
Send ciphertext $< c_i >_A$ to S_A
Send $< sum >_A$ and $< \sigma >_A$
Server A
...
Send ciphertext$< c_i >_B$ to S_B
Send $< sum >_B$ and $< \sigma >_B$
Control center
Mobile users
Server B
Send the VK_i
Send the result and proof
Verifier

Figure 1. System model of the non-interactive and secure data aggregation scheme

Table 1. Definition of notations in the non-interactive and secure data aggregation scheme

Symbols	Definition
U	Mobile user dataset
$< x_i >_A, < x_i >_B$	The shares of x_i
a_i	Random vector of the mask selected by user
$< c_i >_A, < c_i >_B$	The ciphertexts of shared value $< x_i >$
m	Dimension of the private vector
h	Verification vectors
PK	The public key for verification
VK_i	Authentication key for verification
sum	Aggregation result
σ	The proof of computation

4.1.2. Threat Model

In this paper, we assume that mobile users and cloud servers are honest but curious, and the control center is trustworthy.

The mobile users: The mobile users are curious about the private data of others while participating in the aggregation process. However, malicious mobile users may collude with each other to infer private data of others, or use their own private data to collude with cloud servers for the same purpose.

The cloud servers: They may return forged results to the control center, which will lead to wrong results. In addition, the cloud server may also collude with malicious users to infer the private information of other users.

4.1.3. Design Goals

The non-interactive and secure data aggregation scheme proposed in this paper should achieve the following five goals:

1. Input privacy: The data collected by mobile users are sensitive data. These data should be masked before sending to the server; therefore, in this paper we should ensure that the data input is private.
2. Output privacy: Since there are two non-colluding servers, they perform aggregation operations without obtaining the final aggregation result; therefore, in this paper we should ensure the output privacy.
3. Verifiability: The verifier can utilize the verification algorithm to verify whether it is correct when all participants execute the protocol correctly.
4. Non-interactivity: Our scheme guarantees non-interaction between users, which reduces the communication cost. Due to the non-interactivity between users, it will not affect the normal execution of the protocol even if someone drops out during the aggregation process.
5. Efficiency: It is experimentally demonstrated that our aggregation scheme can obtain the aggregation result and verify its correctness with a low computation cost.

4.2. *Non-Interactive and Secure Data Aggregation Scheme*

In this section, the non-interactive and secure data aggregation scheme is described in detail. We suppose that there are n mobile users and each mobile user, O_i, collects a large amount of private data about themselves, such as identity information, personal assets, health status and so on. The data collected by mobile users are called private data, $x_i = (x_{i1}, x_{i2}, ..., x_{im})$, where m is the dimension of private data. Firstly, each O_i uses additive secret sharing to split x_i into two parts: $< x_i >_A$ and $< x_i >_B$, where $< x_i >_A + < x_i >_B = x_i$ mod p. Then, each O_i encrypts $< x_i >_A$ and $< x_i >_B$ using its own generated private key respectively, and also generates the relevant verification keys. Finally, the mobile user sends the ciphertexts to S_A and S_B, respectively. Next, S_A and S_B aggregate the data from n mobile users, then the aggregation results are sent to the control center for analysis. Meanwhile, the public verifier can also verify the aggregation results, as shown in Algorithm 1, and the specific steps are shown in the following subsections.

4.2.1. Key Generation

The system initialization is performed by the key generation center. e: $G_1 \times G_2 \rightarrow G_\tau$ is a bilinear map, where G_1, G_2 and G_τ are multiplicative groups of order p, and p is a large prime, and g and g' are generators of G_1 and G_2, respectively. Later, it publishes the public parameters = $(G_1, G_2, G_\tau, p, g, g')$.

Firstly, a random number δ ($\delta \in Z_p^*$) and the vector $r = (r_1, r_2, ..., r_n)$ are chosen, then it computes $g'' = g'^{\delta}$ based on the public parameters, the vectors $h = (h_1, h_2, ..., h_n)$ and $PK_i = e(g^{r_i}, g')$, where $r_i \in Z_p^*$, $h_i = g^{\delta + r_i}$ and $PK = (PK_1, PK_2, ..., PK_n)$.

4.2.2. Masking of Private Data

This algorithm is used to blind the private vectors of mobile users. Each mobile user, O_i, has private data $x_i = (x_{i1}, x_{i2}, ..., x_{ik}, ..., x_{im})$, where x_{ik} represent the k-th dimension data $(1 < k < m)$. In the following steps, this paper will take the k-th dimension data x_{ik} as an example. Firstly, the mobile user O_i selects a random vector $a = (a_{i1}, ..., a_{ik}, ...a_{im})$, where $a_{ik} \in Z_p^*$, and calculates $< x_{ik} >_A$ and $< x_{ik} >_B$ using the additive secret sharing algorithm. Input a_{ik} into the PRG as a random seed. Subsequently, O_i encrypts $< x_{ik} >_A$ and $< x_{ik} >_B$ respectively, as follows:

$$\begin{aligned} < c_{ik} >_A &=< x_{ik} >_A +PRG(a_{ik}) \\ < c_{ik} >_B &=< x_{ik} >_B -PRG(a_{ik}) \\ VK_{ik} &= PK_i^{x_{ik}} \end{aligned} \tag{2}$$

Then, the mobile user sends the ciphertexts to the cloud server through a secure channel, meanwhile generating public verification keys $VK_i = (VK_{i1}, ..., VK_{ik}, ...VK_{im})$. The mobile user provides the verification keys to the verifier so that the verifier can verify the final result.

Algorithm 1 Non-Interactive and Secure Data Aggregation Scheme

```
begin
Mobile user: Split $x_i$ into $< x_i >_A$ and $< x_i >_B$.
Compute ciphertexts $< c_{ik} >_A =< x_{ik} >_A +PRG(a_{ik})$ and $< c_{ik} >_B =< x_{ik} >_B -PRG(a_{ik})$.
Compute verification keys $VK_{ik} = PK_i^{x_{ik}}$.
Send the ciphertexts and verification keys through a secure channel to server and public verifier, respectively.
Server A: Compute $< sum_k >_A = \sum_{i=1}^n (< x_{ik} >_A +PRG(a_{ik}))$ and $< \sigma_k >_A = \prod_{i=1}^n h_i^{<c_{ik}>_A}$.
Server B: Compute $< sum_k >_B = \sum_{i=1}^n (< x_{ik} >_B -PRG(a_{ik}))$ and $< \sigma_k >_B = \prod_{i=1}^n h_i^{<c_{ik}>_B}$.
Send the $< sum_k >_A$, $< \sigma_k >_A$, $< sum_k >_B$ and $< \sigma_k >_B$.
Control center: Compute $sum_k =< sum_k >_A + < sum_k >_B$ and $\sigma_k =< \sigma_k >_A \cdot < \sigma_k >_B$.
Send the $sum_k$ and $\sigma_k$.
Public verifier: Verify $e(\sigma_k, g') = e(g^{sum_k}, g'') \prod_{i=1}^n VK_{ik}$.
end.
```

4.2.3. Data Aggregation

In the aggregation phase, the server aggregates these ciphertexts received from n mobile users and generates aggregation proofs $< \sigma >_A$ and $< \sigma >_B$, where $< \sigma >_A = (< \sigma_1 >_A, ..., < \sigma_k >_A, ..., < \sigma_m >_A)$, $< \sigma >_B = (< \sigma_1 >_B, ..., < \sigma_k >_B, ..., < \sigma_m >_B)$. S_A and S_B execute the following computations, respectively:

$$\begin{aligned} < sum_k >_A &= \sum_{i=1}^n (< x_{ik} >_A +PRG(a_{ik})) \\ < \sigma_k >_A &= \prod_{i=1}^n h_i^{<c_{ik}>_A} \end{aligned} \tag{3}$$

$$\begin{aligned} < sum_k >_B &= \sum_{i=1}^n (< x_{ik} >_B -PRG(a_{ik})) \\ < \sigma_k >_B &= \prod_{i=1}^n h_i^{<c_{ik}>_B} (1 \le k \le m) \end{aligned} \tag{4}$$

Then, S_A and S_B generate the aggregation results $< sum >_A = (< sum_1 >_A, ..., < sum_k >_A, ..., < sum_m >_A)$ and $< sum >_B = (< sum_1 >_B, ..., < sum_k >_B, ..., < sum_m >_B)$, respectively. Then, S_A and S_B send the aggregation results and proofs to the control center, which will compute $sum_k =< sum_k >_A + < sum_k >_B$ and $\sigma_k =< \sigma_k >_A \cdot < \sigma_k >_B$ $(1 \le k \le m)$ for each k-th aggregation result and proof.

4.2.4. Verification of the Results

The public verifier can check the each k-th final result and proof after receiving the final result from the control center. The verification formula is as follows:

$$e(\sigma_k, g') = e(g^{sum_k}, g'') \prod_{i=1}^{n} VK_{ik} (1 \leq k \leq m) \tag{5}$$

where σ_k and sum_k are provided by the control center, and VK_{ik} is provided by mobile users.

Theorem 1. *According to the verification algorithm, the public verifier can verify the correctness of the final result.*

Proof.

$$\begin{aligned} e(\sigma_k, g') &= e(<\sigma_k>_A \cdot <\sigma_k>_B, g') \\ &= e(\prod_{i=1}^{n} h_i^{<c_{ik}>_A} \cdot \prod_{i=1}^{n} h_i^{<c_{ik}>_B}, g') \\ &= e(g,g')^{\delta \sum_{i=1}^{n}(<c_{ik}>_A + <c_{ik}>_B)} \cdot e(g,g')^{\sum_{i=1}^{n} r_i(<c_{ik}>_A + <c_{ik}>_B)} \\ &= e(g^{sum_k}, g')^{\delta} \cdot \prod_{i=1}^{n} e(g,g')^{r_i(<c_{ik}>_A + <c_{ik}>_B)} \\ &= e(g^{sum_k}, g'') \cdot \prod_{i=1}^{n} VK_{ik} \end{aligned} \tag{6}$$

It can be seen from the above formula that our proposed verification algorithm is correct. □

5. Analysis

We will analyze the non-interactive and secure data aggregation scheme based on input and output privacy, verification algorithm security, robustness analysis and limitations.

5.1. Input and Output Privacy

Input privacy: Our scheme first performs additive secret sharing of private data and then masks it by the random number in the masking phase. The random number, a_{ik}, is generated by the user and kept by himself. It must obtain the blinded value a_{ik} if the cloud server receives the ciphertext of each mobile user and wants to recover the personal information. Assume that $|p|$ is the length of the key, then the probability that the cloud server recovers the private data is $P_r = 2^{-|p|}$. Since the key length is usually very large, the probability is negligible. The server cannot obtain privacy data for the mobile user, and therefore, our scheme guarantees the input privacy.

Output privacy: In our aggregation phase, the server aggregates the ciphertext data from all the mobile users and the output result is a part of the additive secret sharing, which is not the sum of all of the mobile users. Meanwhile, the two servers are non-colluding, honest and curious, so the server does not obtain the final aggregation result and then cannot analyze the data of these mobile users in the aggregation. Moreover, the $<\sigma_k>_A$ and $<\sigma_k>_B$ are computed by the cloud servers and cannot reveal any private information about them; therefore, our scheme guarantees the output privacy.

5.2. Verification Algorithm Security

Theorem 2. *Our proposed verification algorithm is secure under the BDHE assumption.*

Proof. The reason this paper uses the BDHE assumption is that in the verification phase, the user needs to send the verification key $PK_i^{x_i} = e(g^{r_i}, g')^{x_i}$ to the verifier, which is a BDHE problem. We know that the BDHE assumption is that given g, g^{ψ}, $g^{\omega} \in G$, $h \in G_1$, it is difficult to compute ψ, $\omega \in Z_p^*$. Therefore, this paper assumes that there is an adversary P that can work out this difficult matter by using an efficient algorithm, Θ, with a non-negligible advantage, ε. The steps are indicated as below:

The algorithm Θ initializes the system by choosing G_1, G_2 and G_τ as p-order cyclic groups, and g, g' are the generating elements of G_1, G_2, respectively. Firstly, the algorithm Θ chooses ψ , $\omega \in Z_p^*$ randomly, computes $g_1 = g^{\psi}$ and $g_1' = (g'^{\omega})^{\delta}$, and then the common parameters = $(G_1, G_2, G_\tau, p, g, g', g_1, g_1')$ are published. Secondly, $PK_i' = e(g, g')/e(g_1, g_1')$ is computed. Since the algorithm Θ knows the public parameters and VK_{ik}', then it can obtain $e(\prod_{i=1}^{n} g^{x_{ik}}, g') = e(\prod_{i=1}^{n} g_1^{c_{ik}}, g_1) \prod_{i=1}^{n} VK_{ik}'$, where $VK_{ik}' = PK_i'^{c_{ik}}$. Adversary P uses the algorithm Θ to obtain VK_{ik} for each input x_{ik}. Then, adversary P forges the aggregation $result_k'$, where $result_k' \neq result_k$. Then, the algorithm Θ verifies whether $results_k' = result_k$. If the verification passes, it implies that the algorithm Θ fails the challenge. Otherwise, it works out the BDHE problem and obtains the following settlement result:

$$e(g, g')^{\psi\omega} = e(\frac{\delta}{\prod_{i=1}^{n} g^{c_{ik}}})^{\delta(result_k - result_k')^{-1}} \tag{7}$$

From the above equation, it can be seen that our verification algorithm is secure under the BDHE assumption. □

5.3. Robustness Analysis

In the actual application, mobile users in IoT may drop out at any time due to network and other factors. In the previous schemes, the secret sharing recovery algorithm is usually used to restore the mask values of exiting users, which improves the robustness of their scheme. Some methods [12,21] require a large number of mobile users to be online, and their masking formula is $y_u = x_u + PRG(b_u) + \sum_{v \in U: u<v}(PRG(s_{u,v}) - \sum_{v \in U: u>v}(PRG(s_{u,v})$ mod R. The x_u represents the private data of each user and b_u is a random number generated by the user who distributes shares of b_u to the other users, and $s_{u,v}$ is the agreement key between user u and user v. Assume that there are no user drops out, then the cloud server will correctly compute the aggregation result $\sum_{u=1}^{n}(y_u)$, and then the $\sum_{u=1}^{n}(\sum_{v \in U: u<v} PRG(s_{u,v}) - \sum_{v \in U: u>v} PRG(s_{u,v}))$ will be zero. Once a user drops out, a large number of users are needed to recover the shared values b_u and $s_{u,v}$ by the secret reconstruction algorithm to make $\sum_{u=1}^{n}(\sum_{v \in U: u<v} PRG(s_{u,v}) - \sum_{v \in U: u>v} PRG(s_{u,v}))$ zero, so a large number of users need to be online to ensure that the scheme is executed correctly. However, the scheme [20] does not support user drop out. Since the formula of the masking phase is $y_i = x_i + s_i^{-1}(\sum_{j=i+1}^{n} PRG(v_{ij}) - \sum_{j=1}^{i-1} PRG(v_{ij}))$, where x_i represents the private data, s_i is generated by aggregation and v_{ij} is the agreement key between user i and user j. Assume that once the user drops out, then the y_i will not be uploaded to the cloud server, and the formula $(\sum_{j=i+1}^{n} PRG(v_{ij}) - \sum_{j=1}^{i-1} PRG(v_{ij}))$ is not zero, which will produce an incorrect aggregation result.

The scheme in this paper does not require the mobile users to be online all the time. The mobile user sends the ciphertexts $< c_{ik} >_A =< x_{ik} >_A + PRG(a_{ik})$ and $< c_{ik} >_B =< x_{ik} >_B - PRG(a_{ik})$ to the server. Assume that the mobile user O_i drops out, then neither $< c_{ik} >_A$ nor $< c_{ik} >_B$ will be sent to the server, and the data of O_i will not have an influence on the aggregation result. Therefore, our scheme is robust as it will not fail to be performed properly even if some mobile users drop out.

5.4. Limitations

Reviewing the system framework of this paper, we conclude the limitations. First, all participants in this paper are designed to be honest and curious, except for the control center, and the two cloud servers do not collude. The system designed in this paper allows a small number of mobile users to collude with the cloud server, which does not lead to leakage of the private data of mobile users. However, once two cloud servers collude with each other, the privacy of data may be affected. For example, S_A and S_B cooperate to calculate the final aggregation result after the ciphertexts of all mobile users are sent to the cloud server, and then the aggregation result will be maliciously analyzed. Due to the high security assumptions in this paper, we will focus on a single cloud server to solve non-interactive issues in the future.

6. Performance Evaluation

In this section, we evaluate the performance of the non-interactive and secure data aggregation scheme using theoretical and experimental analysis.

6.1. Function Analysis

We compare the functions of the non-interactive and secure data aggregation scheme with the previous ones in Table 2. In [12], the authors use a double-mask to ensure data privacy, efficient aggregation and robustness, which does not realize non-interactivity, verification and result privacy. In [13], the authors also use a double-mask to ensure data privacy, efficient aggregation, robustness and verification, which does not realize non-interactivity and result privacy. In [16], the authors propose a multi-party secure computing framework to ensure data privacy, verification, robustness and result privacy, which does not achieve non-interactivity. In [20], the authors propose one mask to ensure data privacy and verification, which does not achieve non-interactivity, robustness and result privacy. It can be seen from Table 2 that the non-interactive and secure data aggregation scheme of this paper achieves these five functions. Compared with the schemes mentioned above, our scheme has certain advantages.

Table 2. Comparison with previous secure aggregation schemes

Schemes	Data Privacy	Non-Interactity	Verification	Roustness	Result Privacy
[12]	Yes	No	No	Yes	No
[13]	Yes	No	Yes	Yes	No
[16]	Yes	No	Yes	Yes	Yes
[20]	Yes	No	Yes	No	No
ours	Yes	Yes	Yes	Yes	Yes

In Table 3, our scheme is compared with the previous ones in terms of computation cost and communication costs, where $|p|$ is the length of security parameter p, and the number of users and the dimension of private data are n and m, respectively. In terms of computation cost, the scheme in [12] has the highest cost since it requires a large number of secret sharing and secret reconstruction operations, which requires $O(n^2 + mn)$. Xu et al. [13] applied the algorithm in [12], so the time complexity of the user side is the same. In [16], the time complexity of generating ciphertext and signature is $O(m + n)$, and in [20], the main computation cost results from computing the verification key and agreement keys, which requires $O(m + n)$. In our method, the computational cost is independent of the number of users and it is only relevant to the size of data vector m, so the cost is the smallest.

In terms of communication cost, the schemes in [12,13] include key agreement and secret sharing, which require $6(n-1)m|p|$. In [16], the users only need to upload their own encrypted data, which requires $m|p|$. However, the non-interactive and secure data aggregation scheme makes the communication overhead between users zero. In our scheme, the ciphertexts and verification keys are sent to the server and the verifier, respectively. Then, the communication cost between the mobile user and the server is $2m|p|$, and with the verifier is $m|p|$, so our proposed scheme has the lowest communication cost.

Table 3. Comparison of the computation and communication costs.

Computational Cost Comparison				**Communication Cost Comparison**		
Schemes	**User**	**Server**	**Verifier**	**User–User Communication**	**User–Server Communication**	**User–Verifier Communication**
[12]	$O(n^2 + mn)$	$O(mn^2)$	-	$6(n-1)m\|p\|$	$7m\|p\|$	-
[13]	$O(n^2 + mn)$	$O(mn^2)$	$O(m)$	$6(n-1)m\|p\|$	$8m\|p\|$	-
[16]	$O(m + n)$	$O(mn)$	$O(mn)$	$m\|p\|$	$3m\|p\|$	-
[20]	$O(m + n)$	$O(mn)$	$O(mn)$	$2(n-1)m\|p\|$	$2m\|p\|$	$m\|p\|$
ours	$O(m)$	$O(mn)$	$O(mn)$	-	$2m\|p\|$	$m\|p\|$

6.2. Experimental Results

We carried out a series of experiments to show the feasibility and efficiency of the scheme. The experiments are based on python language implementation. In the verification part, we used the bilinear map pypbc library, which is a python wrapper for the PBC library, and set p as 1024 bits. All the experiments depend on the computer with Intel(R) Core(TM) i5-10210U CPU @ 1.60 GHz 2.11 GHz 16.0 GB RAM, and run on the Ubuntu 18.04 operating system.

The first experiment was executed to evaluate the efficiency of each sub-algorithm in our scheme. In this experiment, our scheme has four sub-algorithms, including key generation, masking of private data, data aggregation and verification of the results. We defined the following parameters to represent the corresponding operation: T_{exp} , T_{pair} and T_{mm} for modular exponentiation, bilinear pairing and modular multiplication operations, which are 0.16, 23 and 0.02 ms, respectively.

In the key generation phase, the algorithm chooses a random vector $r \in z_p^*$, and computes g'', h_i and PK_i. Computing g'' and h_i requires one modular exponentiation and n modular exponentiation operations respectively (where $h_i = g^{\delta + r_i}(1 < i < n)$), while computing PK_i requires n modular exponentiations and one bilinear pairing operation. From this phase, we conclude that the total time cost is $(2n+1)T_{exp} + T_{pair}$.

In the masking phase, the main overhead of the mobile user is to compute verification keys VK_i. Computing the verification keys require m modular exponentiation operations (where $VK_{ik} = PK_i^{x_{ik}}(1 < k < m)$), so the total computation overhead is mT_{exp}. From this phase, we conclude that the computation overhead of the masking phase has a linear relationship with the data size m.

In the aggregation phase, the servers calculate the aggregation values $< sum >_A$ and $< sum >_B$, respectively. However, the servers' main time overhead is to calculate the generation proof $< \sigma >_A$ and $< \sigma >_B$, which requires $2mn$ modular exponentiation and $2(n-1)$ modular multiplication operations. Therefore, the computational cost is $2mnT_{exp} + 2(n-1)T_{mm}$. From this phase, it displays that the computational cost of the server in the aggregation phase is related to the mobile user n and the size of data m.

In the verification phase, the verifier receives the final result and proofs from the control center, which requires $2m$ modular exponentiation, two bilinear pairing and n modular multiplication operations. Therefore, the total time cost is $nT_{mm} + 2T_{pair} + 2mT_{exp}$. This phase displays that the computation cost of the verification phase is related to the mobile user n and the size of data m.

In the proposed aggregation scheme, the mobile users were set from 100 to 600, while the private data dimension is fixed at $m = 10$. As shown in Figure 2, we compared the relationship between the computational cost of each phase and the number of users n. In the keyGen phase, the main computation overhead is to compute the key of the public verifier, which requires $(2n+1)T_{exp} + T_{pair}$. In the masking phase, the main computational cost is to calculate the verification key on the client side, which only requires mT_{exp} and is independent of the number of users. In the verification phase, the verifier mainly verifies the Equation (5), which requires $nT_{mm} + 2T_{pair} + 2mT_{exp}$. As shown in this paper, n is set from 100 to 600, and m is only set to 10. Therefore the computational cost of key generation is the largest. We can see the computation overhead in the aggregation phase is between 324 and 1944 ms, while that in the verification phase is between 51.2 and 61.2 ms.

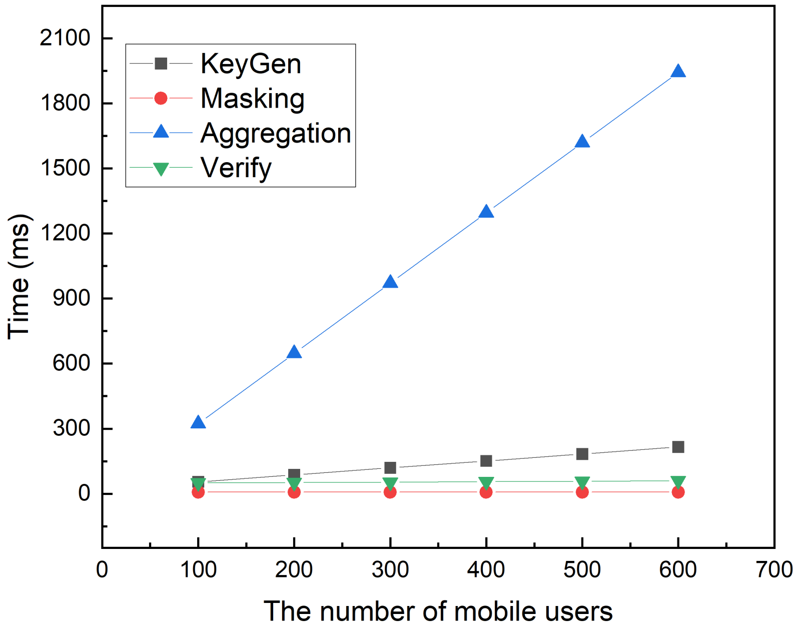

Figure 2. The time cost of each phase varies with the number of mobile users.

Then, we change the data dimension when the number of mobile users is fixed $(n = 10)$, and it can be seen that the overhead of each phase increases with the dimensions. It can be seen from Figure 3 that the computational overhead of the key generation phase is independent of the data dimensions when the dimensions are set from 100 to 600. In the masking phase, the time cost of the user is between 16 and 96 ms. In the aggregation phase, the computational cost of the server is between 320.04 and 1920.24 ms, while the computational cost of the verification phase is between 62.02 and 142.02 ms.

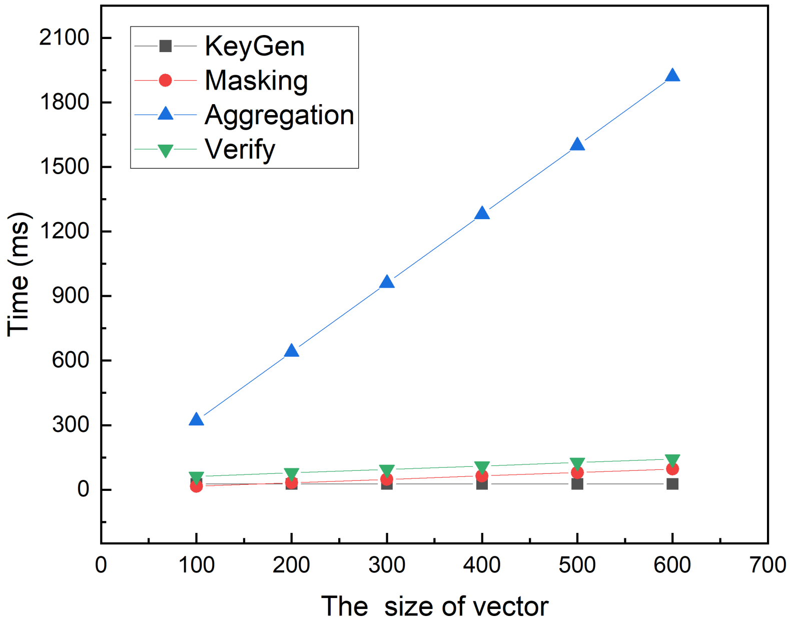

Figure 3. The time cost of each phase changes with the dimensionality of the data.

In the experiments, as shown in Figure 4, we compared the time cost of the user-side of different schemes, which varies with the dimension of the data. Specifically, the number of users is set at $n = 10$, and the data dimensions vary from 100 to 500. The experiments suppose an ideal situation where no mobile users drop out during the aggregation phase. Since the schemes in [12,13] use secret sharing and the Diffie-Hellman key agreement protocol, the time cost of their schemes is relatively high. However, Ma et al. [16] employ proxy re-encryption and aggregation signatures, which contain modular exponentiation operations, so the computation cost is not large. The scheme in [20] adopts key agreement and the aggregation coefficient to mask the private data, which only contains a small number of modular exponentiation and modular multiplication operations, so the time cost is not large. In the non-interactive and secure data aggregation scheme, the main overhead of mobile users is to calculate the verification key, which only contains a small number of modular exponentiation operations. Therefore, we can see that the computation cost of the other three schemes is much higher than ours.

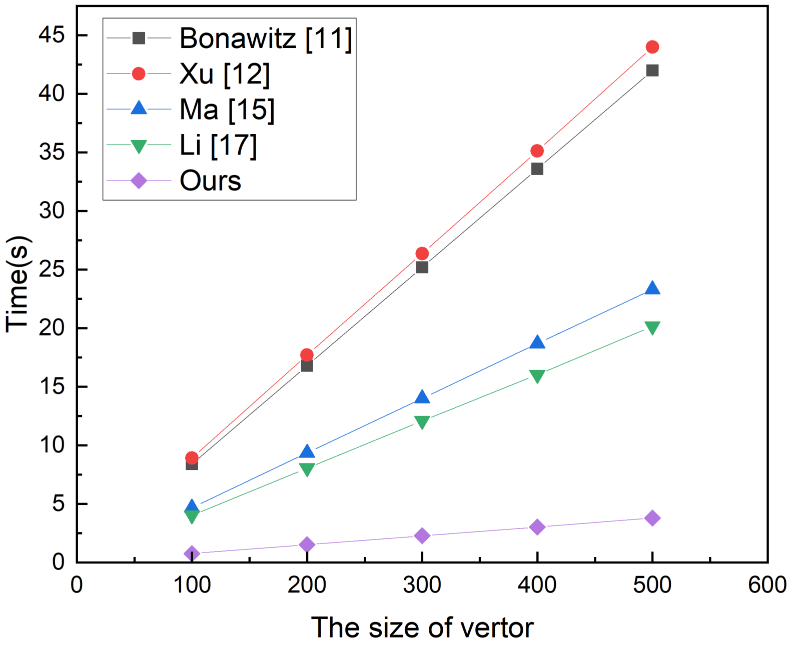

Figure 4. The time cost of the user-side of different schemes varies with the dimensionality of the data.

7. Conclusions

In this paper, we proposed a non-interactive and secure data aggregation scheme with additive secret sharing for IoT. The mobile users adopt additive secret sharing to split the private data into two parts and then mask them. Then, the ciphertexts are sent to servers for aggregation. The proposed scheme not only protects the private data of mobile users but also the privacy of the final aggregation results. Moreover, it realizes non-interaction between users and supports users offline. Security analysis showed that the proposed scheme protects the privacy of inputs and outputs. Finally, the experimental results show that this scheme is effective and has less computation and communication costs than the previous schemes.

In future work, we will focus on the following two issues: (1) Mobile user identity privacy: in the research of IoT data security aggregation, most of the papers did not protect the identity of mobile users, and we think that the identity of mobile users is as important as private data, and (2) Consider a single cloud service: in future work, we will use a single server to design a non-interactive aggregation scheme to reduce the computation and communication overhead of mobile users.

Author Contributions: Conceptualization, Y.F., Y.R., G.F., X.Z. and C.Q.; methodology, Y.F. and Y.R.; software, Y.F.; validation, G.F., X.Z. and C.Q.; formal analysis, Y.R.; investigation, Y.F.; resources, Y.F., Y.R., G.F. and X.Z.; data curation, Y.F.; writing—original draft preparation, Y.F.; writing—review and

editing, Y.F.; visualization, Y.R.; supervision, Y.R.; project administration, Y.R.; funding acquisition, Y.R. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (62072295) and the Natural Science Foundation of Shanghai (20ZR1419700).

Acknowledgments: The authors would like to thank the anonymous reviewers for their helpful comments.

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

References

1. The Mobile Economy 2020. Available online: https://www.gsma.com/mobileeconomy/wp-content/uploads/2020/03/GSMA_MobileEconomy2020_Global.pdf (accessed on 5 March 2020).
2. Mandal, K.; Gong, G.; Liu, C. Nike-based fast privacy-preserving highdimensional data aggregation for mobile devices. In *IEEE T Depend Secure*; Technical Report; University of Waterloo: Waterloo, ON, Canada, 2018; pp. 142–149.
3. Jian, S.; Chen, W.; Li, T.; Xiao, F.C.; Huang, X.Y.; Zhan, Z.H. Secure data uploading scheme for a smart home system. *INS* **2018**, *453*, 186-197. [CrossRef]
4. Wu, Z.D.; Liang, B.; You, L.; Jian, Z.H.; Li, J. High-dimension space projection-based biometric encryption for fingerprint with fuzzy minutia. *Soft Comput.* **2016**, *20*, 4907–4918. [CrossRef]
5. He, D.B.; Neeraj, K.; Shen, H.; Jong-Hyouk, L. One-to-many authentication for access control in mobile pay-TV systems. *Sci. China Inform. Sci* . **2016**, *59*, 1–14. [CrossRef]
6. Xu, C.; Xie, X.; Zhu, L.H. PPLS: A Privacy-Preserving Location-Sharing Scheme in Vehicular Social Networks. *Sci. China Inform. Sci.* **2020**, *063*, 163–173. [CrossRef]
7. Zhang, Y.H.; Zheng, D.; Deng, R.H. Security and privacy in smart health: Efficient policy-hiding attribute-based access control. *IEEE Internet Things J.* **2018**, *5*, 2130–2145. [CrossRef]
8. Sattler, F.; Wiedemann, S; Müller, K.R.; Samek, W. Robust and communication-efficient federated learning from non-iid data. *IEEE Trans. Neural Netw. Learn.* **2019**, *31*, 3400–3413. [CrossRef]
9. Chen, Y.; Sun, X.Y.; Jin, Y.C. Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation. *IEEE Trans. Neural Netw. Learn.* **2019**, *31*, 4229–4238. [CrossRef]
10. Liu, X.Y.; Li, H.W; Xu, G.W.; Lu, R.X.; He, M. Adaptive privacy-preserving federated learning. *Peer Peer Netw. Appl.* **2020**, *13*, 2356–2366. [CrossRef]
11. Hao, M.; Li, H.W.; Luo, X.Z.; Xu, G.W.; Yang, H.M.; Liu, S. Efficient and privacy-enhanced federated learning for industrial artificial intelligence. *IEEE Trans. Industr. Inform.* **2019**, *16*, 6532–6542. [CrossRef]
12. Keith B.; Vladimir, I.; Ben, K.; Antonio, M.; McMahan, H.B.; Patel, S.; Ramage, D.; Segal, A.; Seth, K. Practical secure aggregation for privacy-preserving machine learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, 30 October 2017; pp.1175–1191.
13. Xu, G.W.; Li, H.W.; Liu, S.; Yang, K.; Lin, X.D. Verifynet: Secure and verifiable federated learning. *IEEE Trans. Inf. Forensics Secur.* **2019**, *15*, 911–926. [CrossRef]
14. Yang, W.Q.; Liu, B.; Lu, C.L.; Yu, N.H. Privacy preserving on updated parameters in federated learning. In Proceedings of the ACM Turing Celebration Conference, Hefei, China, 22 May 2020; pp.27–31.
15. Kalikinkar, M.; Gong, G. Privfl: Practical privacy-preserving federated regressions on high-dimensional data over mobile networks. In Proceedings of the 2019 ACM SIGSAC Conference on Cloud Computing Security Workshop, London, United Kingdom, 11 November 2019; pp. 57–68.
16. Xu,M.;Ji, C.M.; Zhang, X.Y.; Wang, J.F.; Li, J.; Li, K.C.; Chen, X. Secure multiparty learning from the aggregation of locally trained models. *J. Netw. Comput. Appl.* **2020**, *167*, 1084–8045. [CrossRef]
17. Al-Zubaidie, M.; Zhang, Z.; Zhang, J. REISCH: Incorporating Lightweight and Reliable Algorithms into Healthcare Applications of WSNs. *Appl. Sci.* **2020**, *10*, 2007. [CrossRef]
18. Edemacu, K.; Kim, J.W. Multi-Party Privacy-Preserving Logistic Regression with Poor Quality Data Filtering for IoT Contributors. *Electronics* **2021**, *10*, 2049. [CrossRef]
19. Ming, Y.; Zhang, X.Y.; Shen, X.Q. Efficient privacy-preserving multi-dimensional data aggregation scheme in smart grid. *IEEE Access* **2019**, *7*, 32907–32921. [CrossRef]
20. Li, T.; Gao, C.Z.; Jiang, L.L.; Witold P.; Shen, J. Publicly verifiable privacy-preserving aggregation and its application in IoT. *J. Netw. Comput. Appl.* **2019**, *126*, 39–44. [CrossRef]
21. Jiang, Y.; Zhao, B.W.; Tang, S.H.; Wu, H.T. A verifiable and privacy-preserving multidimensional data aggregation scheme in mobile crowdsensing. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4008. [CrossRef]
22. Yao, A.C.C. How to generate and exchange secrets. In Prpceedings of the 27th Annual Symposium on Foundations of Computer Science (sfcs), Toronto, Canada, 27-29 October 1986; pp.162–167.
23. Qiu, S.; Wang, B.Y.; Li, M.; Liu, J.Q.; Shi, Y.F. Toward practical privacy-preserving frequent itemset mining on encrypted cloud data. *IEEE Trans. Cloud Comput.* **2017**, *8*, 312–323. [CrossRef]

24. Verykios, V.S.; Bertino, E.; Fovino I.N.; Provenza, L.P.; Saygin, Y.; Theodoridis, Y. State-of-the-art in privacy preserving data mining. *ACM Sigmod Rec.* **2004**, *33*, 50–57. [CrossRef]
25. Rakesh, A.; Ramakrishnan, S. Privacy-preserving data mining. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, New York, NY, USA, 16 May 2000; pp. 439–450.
26. Wenliang D.; Mikhail j.a.; et al. Privacy-preserving cooperative scientific computations. In *csfw*, Cape Breton, NS, Canada, 11-13 June 2001; Volume 1, p. 273.
27. Liu, L.; Su, J.; Liu, X.; Chen, R.; Huang, K.; Deng, R.H.; Wang, X. Toward highly secure yet efficient KNN classification scheme on outsourced cloud data. *IEEE Internet Things J.* **2019**, *6*, 9841–9852. [CrossRef]
28. Ma, X.; Chen, X.; Zhang, X. Non-interactive privacy-preserving neural network prediction. *Inf. Sci.* **2019**, *481*, 507–519. [CrossRef]
29. Manuel, B.; Silvio, M. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM J. Comput.* **1984**, *13*, 850–864. [CrossRef]
30. Andrew, C.Y. Theory and application of trapdoor functions. In Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (SFCS), Chicago, IL, USA, 3–5 November 1982; pp. 80–91. [CrossRef]
31. Dan, B.; Craig, G.; Ben, L.; Hovav S. Aggregate and verifiably encrypted signatures from bilinear maps. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Berlin/Heidelberg, Germany, 13 May 2003; pp. 416–432.
32. Chen,X.F.; Li, J.; Huang, X.Y.; Ma, J.F.; Lou, W.J. New publicly verifiable databases with efficient updates. *IEEE Trans. Dependable Secur. Comput.* **2014**, *12*, 546–556. [CrossRef]
33. Li, J.; Huang, X.Y.; Li, J.W.; Chen, X.F.; Xiang, Y. Securely outsourcing attribute-based encryption with checkability. *IEEE Trans. Parallel. Distrib. Syst.* **2013**, *25*, 2201–2210. [CrossRef]
34. Chen, X.F.; Li, J.; Weng, J.; Ma, J.F.; Lou, W.J. Verifiable computation over large database with incremental updates. *IEEE Trans. Comput.* **2015**, *65*, 3184–3195. [CrossRef]
35. Li, J.; Chen, X.F.; Li, M.Q.; Li, J.W.; Lee, P.P.; Lou, W. Secure deduplication with efficient and reliable convergent key management. *IEEE Trans. Parallel Distrib. Syst.* **2013**, *25*, 1615–1625.
36. Jia, K.; Li,H.W.; Liu, D.X.; Yu, S. Enabling efficient and secure outsourcing of large matrix multiplications. In Proceedings of the 2015 IEEE Global Communications Conference (GLOBECOM), San Diego, CA, USA, 6–10 December 2015; pp. 1–6. [CrossRef]
37. Zhang, X.Y.; Jiang, T.; Li, K.C.; et al. New publicly verifiable computation for batch matrix multiplication. *Inf. Sci.* **2019**, *479*, 664–678. [CrossRef]