Robust and Lightweight Data Aggregation with Histogram Estimation in Edge-Cloud Systems

Su, Yuan; Li, Jiliang; Su, Zhou; Meng, Weizhi; Yin, Hao; Lu, Rongxing

Published in:
IEEE Transactions on Network Science and Engineering

Link to article, DOI:
10.1109/TNSE.2024.3352734

Publication date:
2024

Document Version
Peer reviewed version

Robust and Lightweight Data Aggregation with Histogram Estimation in Edge-Cloud Systems

Yuan Su, Jiliang Li, Jiahui Li, Zhou Su, *Senior Member, IEEE*, Weizhi Meng, *Senior Member, IEEE*, Hao Yin, and Rongxing Lu, *Fellow, IEEE*

Abstract—Secure aggregation based on masked encryption is a crucial technique for data collection in the Internet of Things (IoT) as it employs a lightweight style to enable global data aggregation while protecting individual data. However, network instability makes the design of such schemes more complex as dropout-resiliency is required, where the overheads substantially increase with growing dropped users. Moreover, existing methods primarily concentrate on aggregation and fail to support complex data analysis, such as histogram estimation. This paper proposes a Robust and Lightweight Data Aggregation (RLDA) scheme in edge-cloud systems. RLDA leverages the offline/online paradigm to achieve robust data aggregation, where edge nodes are introduced to assist verifiable key generation offline and data aggregation and key recovery online. RLDA decouples keys of dropped and surviving users so that it can reduce the overhead by always recovering the keys of surviving users rather than reconstructing the keys of growing dropped users. To achieve secure histogram estimation, we design two recoverable aggregation algorithms that support the transformation between vector and single value, and additionally support multidimensional data aggregation. We prove the security and dropout-resiliency of RLDA. The performance shows that RLDA significantly reduces the overhead with growing dropped users.

Index Terms—Robust aggregation, Secure histogram estimation, Multidimensional data aggregation, Edge computing.

1 Introduction

As the scale of Internet of Things (IoT) devices grows and expands, the world is becoming increasingly connected, and users' lifestyles become more convenient and intelligent [1, 2]. The ubiquitous IoT devices are able to share a large amount of collected data with cloud server (CS), which can conduct data analysis to provide better service through machine learning [3, 4]. For example, the CS in smart grids can create a full picture of energy use across areas ranging from villages to cities by collecting data of smart meters, thus optimizing energy usage and efficiently distributing energy relative to load. In the Internet of Vehicles (IoV), the CS collects and analyzes the traffic data from vehicles to obtain global traffic information, and thus making an intelligent route plan for users. The CSs are only interested in learning aggregate statistics of users' devices, but often attempt to collect private data from users. However, the caches of users' private data poses severe privacy and security risks: private data may be disclosed by motivated attackers [5], or be misused for profit by cloud servers [6].

To learn aggregate statistics while protecting private data of users, secure aggregation method allows users to encrypt data and upload it to CS, where CS can aggregate the encrypted data, and decrypt it to learn the final aggregated outcome. In the existing works, homomorphic encryption, differential privacy, and masked encryption are primarily leveraged to achieve secure aggregation. The homomorphic encryption brings high computational complexity and heavy overhead of data expansion, which burdens the constrained IoT users [7–12]. Differential privacy [13] provides resilient aggregation, and could moderately sacrifice efficiency and accuracy [14–17]. Different from homomorphic encryption and differential privacy, masked encryption uses lightweight symmetric encryption to protect data confidentiality and provide lightweight data aggregation [18–25]. The masked encryption based schemes [18–20] utilize pairwise random-seed agreement between IoT users to generate encryption keys, which are then used to encrypt individual data using an additive key structure that permits secure aggregation at the CS by allowing key cancellation. By grouping users, these schemes [21–25] directly use Shamir secret sharing [26] to share data with users in the next group and enable data aggregation of data sharings for each group. The CS can finally reconstruct the aggregated data by sufficient data sharings of users in the final group.

However, IoT users could be disconnected in the intermittent and unstable network. In order to guarantee correct and precise aggregation, the users in the groups and the CS need to communicate with surviving users in multiple rounds to recover keys or data sharings of dropped users. This raises the first issue: **the growing number of dropped users leads to an increased number of reconstructed keys,**

- *This work was supported in part by the NSFC under Grant No.U20A20175, Grant No.U1808207, Grant No.62102305, Grant No.92067206 and in part by the Fundamental Research Funds for the Central Universities.*
- *Yuan Su, Jiliang Li and Zhou Su are with the School of Cyber Science and Engineering, Xi'an Jiaotong University, Xi'an, China (e-mail: suyuan@stu.xjtu.edu.cn, jiliang.li@xjtu.edu.cn, zhousu@ieee.org).*
- *Jiahui Li is with School of Mathematics and Statistics, Xidian University, Xi'an, China (e-mail: jiahuili@stu.xidian.edu.cn).*
- *Weizhi Meng is with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, 2800 Copenhagen, Denmark (e-mail: weme@dtu.dk).*
- *Hao Yin is with the Research Institute of Information Technology (RIIT), Tsinghua University, Beijing 100084, China (e-mail: h-in@mail.tsinghua.edu.cn).*
- *Rongxing Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: rlu1@unb.ca).*

which increases the overhead of reconstruction at the users and CS. Instead of recovering the keys of dropped users through multi-rounds communications, we focus on one-shot key recovery of surviving users. To achieve this goal, we introduce an edge-cloud framework for data collection, which can deliver higher performance or lower cost by processing partial data aggregation at the edge nodes (ENs). The ENs are responsible for managing a cluster of users and executing partial aggregation, key preparation, and recovery procedures with minimal overhead. Thus, one-shot key recovery can be performed through aggregated sub-keys of ENs.

Moreover, secure aggregation schemes [18–25] based on masked encryption are primarily designed to achieve secure aggregation, and do not support complex data analysis, such as histogram estimation [27, 28]. As the histogram estimation offers insights into the range, center, dispersion, and shape of data, as well as the identification of outlier values or intervals, it is vital for data analysis in transportation (e.g., traffic estimation of specific area), energy sectors (e.g., optimize energy usage), etc. This raises the second issue: **Lack of histogram estimation makes it difficult to analyze data distribution characteristics effectively**. Therefore, it is imperative to achieve histogram estimation in secure aggregation for edge-cloud scenarios. This paper extends to support secure histogram estimation by designing Recoverable Aggregation (RecAgg) algorithms, which allow users to submit data vectors securely. By this way, CS can construct the histogram of a set of values from users but without ever learning which data a particular user owns.

In this paper, we design a Robust and Lightweight Data Aggregation (RLDA) scheme with secure histogram estimation. RLDA provides lightweight aggregation in an onilne/offline paradigm, where a steady supply of encryption keys can be generated in a verifiable way to support key recovery in presence of dropped users during the offline phase. As a result, one such encryption key is available to process the request online in an efficient manner when an encryption request is received. Furthermore, RecAgg algorithms are designed to support histogram estimation. Our contributions are summarized as follows.

- We propose a robust and lightweight data aggregation scheme using the online/offline paradigm, which provides verifiable key generation in the offline phase, and robust data aggregation in the online phase. By one-shot key recovery, RLDA can efficiently recover the decryption keys for correct and precise aggregation.
- To achieve efficient histogram estimation, we propose two RecAgg algorithms, which can aggregate a vector into a single value, and de-aggregate a single value to retrieve the original vector, conversely. Based on the RLDA and the RecAgg algorithms, the secure histogram estimation is designed in a lightweight way. Additionally, multidimensional data aggregation is also supported in our designs.
- We formally prove the correctness and security of our RLDA scheme, and also prove that RLDA provides robust and privacy guarantee by analyzing information entropy. We implement our RLDA and conduct a comprehensive performance comparison with existing schemes. Experimental results demonstrate the feasibility and efficiency of our RLDA in presence of growing dropped users.

The remaining sections of this paper are organized as follows. We provide a comprehensive review of related works in Section 2, followed by the introduction of the system model and threat model in Section 3. In Section 4, we present our proposed RLDA scheme. Section 5 discusses the construction of secure histogram estimation, along with two RecAgg schemes. The security is proved in Section 6, and performance evaluation of our RLDA is shown in Section 7. Finally, we summarize our work in Section 8.

2 Related work

Many secure aggregation schemes with various functionality are proposed to adapt to diverse scenarios, such as federated learning, smart grid, IoT, IoV, etc. The main methods are divided into the following categories: homomorphic encryption, differential privacy and masked encryption based aggregation.

Homomorphic encryption. Secure aggregation built on homomorphic encryption can achieve precise aggregation results and functions, such as arithmetic operations on the encrypted data [7, 8], robust guarantee [9], multi-types data aggregation [10], etc. Based on the double trapdoor cryptosystem, an efficient and privacy-preserving scheme is proposed to achieve secure aggregation and function query [7]. To alleviate the burden of the control center, ElGamal-OU is proposed to improve the efficiency of processing ciphertexts, and achieves privacy protection and function query [8]. SEAR employs the trusted execution environment and public encryption to achieve secure model aggregation for Byzantine-Robust Federated Learning [9]. Observing that edge nodes always generate different types of data that will be further processed in-depth analysis, VPMDA is proposed to collect multi-type data [10]. Focusing on protecting edge data in intelligent transportation systems, the decoding function is constructed by a partially encrypted secure multi-party broadcast computation algorithm, which also shares the gradients among the local models with $O(n)$ time complexity [11]. Unfortunately, homomorphic encryption based secure aggregation schemes provide reliable results and multi-function, but incur heavy overhead on constrained users' devices in terms of computational complexity and data expansion.

Differential privacy. Differential privacy [13] is a noise-based technique that adds random noise to users' raw data to prevent data leakage, while the noise does not have a significant impact on the aggregated results. The goal of differential privacy is to provide reasonable data analysis results while protecting users' privacy [14–17]. However, the added random noise may blur some data features, leading to a decrease in the accuracy of analysis results. This accuracy loss can be mitigated by increasing the noise magnitude, but it also strengthens privacy protection and makes the analysis results more uncertain. Thus, appropriate privacy parameters and noise magnitudes need to be chosen based on specific application scenarios and data

sensitivity, to meet the balance between privacy protection needs and data analysis requirements.

Masked encryption. The masked encryption based data aggregation employs the symmetric homomorphic encryption [29, 30] to achieve secure data aggregation. Based on double-masking technique, SecAgg designs a secure and dropout-tolerant aggregation protocol, where the self-mask is generated by the users and the pairwise-mask is generated between each pair users [18]. Then each user sends data encrypted by the masks using an additive structure such that the summation of additive masks can cancel out when the encrypted data is aggregated. Based on the framework of SecAgg, VeriFL is proposed to achieve verifiable aggregation by linearly homomorphic hash and commitment [19]. TubroAgg specializes in secure aggregation for federated learning [21]. Through random grouping of users and pairwise key agreement with adjacent group users, TubroAgg reduces communication costs, and also provides robust guarantee by Lagrange coding [31]. The communication complexity is reduced from $O(n^2)$ to $O(nlogn)$ with only $O(n)$ rounds by circular communication.

In [22], LightSecAgg is proposed to protect the privacy of each user's individual model while supporting global aggregation. Each EN partitions a key vector of high dimension into a key vector of low dimension, and encodes the low dimension vector by the maximum distance separable code to prepare least but sufficient key shares for other users, allowing efficient key recovery in presence of dropped users. Thus, LightSecAgg provides robustness against dropped nodes and privacy against colluded nodes. Based on Shamir's secret sharing, CodedPaddedFL and CodedSecAgg provide straggler resiliency and robustness [23]. SwiftAgg uses group partition to minimize the communication cost in presence of dropped users, and employs secret sharing to provide data confidentiality [24]. Besides, an information theoretic secure aggregation scheme is presented in [25], which supports two rounds of secure aggregation and dropped users.

Although existing masked encryption schemes that based on secret sharing and coding techniques can achieve secure data collection, these schemes incur communication and computation overhead in presence of growing dropped users, and fail to support histogram estimation. Therefore, we propose a robust and lightweight aggregation scheme with secure histogram estimation in edge-cloud systems.

3 Problem Statement

The goal of our RLDA is to enable the CS to gain aggregated value and histogram estimation of data among all users, while leaking as little private data of individual user to the CS as possible. Our proposed RLDA works on a verifiable online/offline model, and provides strong robust guarantee and privacy guarantee, i.e., simultaneously achieves robust guarantee of $d = \frac{k}{2} - 1$ and privacy guarantee of $t = \frac{k}{2}$. Some notations are defined in Table 1.

3.1 System Model

As shown in Fig. 1, our RLDA scheme consists of a CS, a small number of ENs and a large number of users. At each time epoch, U_i holds a private data m_i in a set of data values $\mathcal{D} = \{0, \cdots, B-1\}$, the goal is to allow CS to obtain the aggregated data of all users (Section 4) and acquire the histogram estimation of all user's data (Section 5), in a way that leaks as little private data of individual user to the CS as possible. The work mode of entities in our scheme is shown as follows.

- **Users (Us)**. The proposed scheme involves n distributed users along with resource-constrained IoT devices. At every time epoch, users collect or generate private data, which is then transmitted to ENs for further aggregation.
- **Edge nodes (ENs)**. The ENs are responsible for partially aggregating the private data of all users located in its region, and sending the aggregated data to the CS for analysis. Suppose there are k ENs in total, and each user resides in only one region.

TABLE 1: Notations

Notations	Semantics
$[1, n]$	The set $\{1, 2, \cdots, n\}$
$E[ij]$	The j-th element of i-th row of matrix E
E_R	The submatrix of the rows $i \in R$ of matrix E
$m \in \mathcal{D}$	Individual private data ($\lvert\mathcal{D}\rvert = B$)
k, n	The number of ENs and Us
d, t, e	Dropout, collusion and recovery threshold
F	Pseudorandom function
α	PRF key
a	Encryption key
β	Unique value of EN
s_{ij}, s_j	EN_j's shares of key a_i, EN_j's sub-mask
cm, Δ	Commitment and commitment set
c_j, C_j, C	Ciphertexts
$\mathcal{U}$	The set of surviving users
$R, \{b_i\}_{i=1}^n$	Auxiliary information
$T_i (i \in [0, n])$	Intermediate parameters

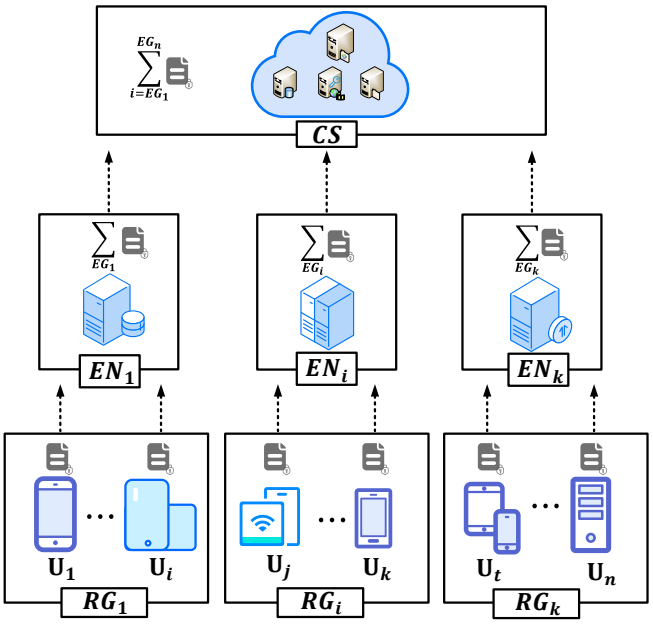


Fig. 1: System model.

Let $RG_1, \cdots, RG_k$ be k sets of users where RG_i is maintained by EN_i, all these k sets are disjoint and satisfy $|RG_1| + \cdots + |RG_k| = n$.

- **Cloud server (CS)**. The CS is a cloud server and aims to acquire data from users for further analysis.

3.2 Threat Model

In the proposed RLDA, authenticated and encrypted point to point (P2P) channels need to be established for each pair of users and ENs, and also for each pair of ENs and CS [6]. In the network, users and ENs may potentially drop out during the data collection for various reasons such as having unreliable and unstable communication connections. Our RLDA considers a threat model where the CS, ENs, and users are honest but curious [32] meaning that all parties will faithfully execute the protocol but are curious about the private information of honest parties. The following security properties are defined, and then the security model are formalized.

- **Completeness**. If all users and ENs honestly follow the scheme, then the CS can correctly obtain the aggregated data and histogram estimation.
- **Robustness to dropped ENs**. In our RLDA scheme, we assume that there are at most d dropped ENs, and there are at least $k - d$ surviving ENs after potential dropouts. Our RLDA scheme guarantees that the CS can correctly decrypt the aggregated data of all surviving users, even if up to d dropped ENs. We have no limit on the number of dropped users, if all users have dropped in a region, the corresponding EN just does not send messages to the CS.
- **Privacy against colluded users and ENs**. Up to t (out of k) ENs can collude together as well as with the CS to attempt to infer the individual private data. In the proposed RLDA scheme, the EN also could collude with the users in its region. In the worst case, all users in the region of t ENs could collude together when these t ENs form a coalition of conspirators. Our RLDA is required to guarantee that nothing can be inferred except for the aggregated data, even in the worst case (up to t ENs collude together where all users in the region of t ENs also conspire jointly). Moreover, arbitrary numbers of users in a region can collude with the corresponding EN to gather the private data of honest users.

Definition 1. *For any pair of robust guarantee d and privacy guarantee t satisfying $d + t < k$, the RLDA scheme can simultaneously achieve robust aggregation against up to any d dropped ENs and protect privacy against up to any t colluding ENs.*

We first recap the definition of pseudorandom function [33].

Definition 2 (Pseudorandom Function)**.** *Let $PRF : \mathcal{K} \times \mathcal{V} \to \mathcal{W}$ be a deterministic polynomial-time algorithm, with key space $\mathcal{K}$, domain $\mathcal{V}$ and range $\mathcal{W}$. For any $\beta \in \{0, 1\}$, we define the experiment IND_β^{PRF} as shown in Fig. 2, where*

$$\mathcal{O}_{PRF}(t) = \begin{cases} PRF(k,t) & if \quad \beta = 0, \\ RF(t) & if \quad \beta = 1 \end{cases}$$

with $RF(t)$ denotes a random function.
A pseudorandom function PRF is secure, if for any PPT adversary $\mathcal{A}$, of which the advantage to distinguish the PRF from random function is negligible, i.e.,

$$\begin{aligned} Adv_{PRF,\mathcal{A}}^{IND}(\lambda) &= |Pr[IND_0^{PRF}(\lambda, \mathcal{A})] - Pr[IND_1^{PRF}(\lambda, \mathcal{A})]| \\ &\leq negl(\lambda), \end{aligned}$$

where $negl(\cdot)$ is a negligible function.

$IND_\beta^{PRF}(\lambda, \mathcal{A})$
$k \leftarrow \mathcal{K}$
$b \leftarrow \mathcal{A}^{\mathcal{O}_{PRF}(\cdot)}(1^\lambda)$
$Output : b$

Fig. 2: Security game for PRF.

Then, we formalize a security model by a secure game between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$ to demonstrate that $\mathcal{B}$ can break a difficult problem using $\mathcal{A}$ as a subroutine if $\mathcal{A}$ can break the RLDA. The game proceeds as follows.

- Setup. The challenger $\mathcal{B}$ generates PRF keys $\{\alpha_j\}_{j=1}^n$ for all n users, and sends $\mathcal{A}$ the PRF keys of a set of users S', which are the corrupted users randomly selected by the adversary $\mathcal{A}$. The set U consists of all n users.
- Query. When receiving an encryption query (j, m_j, r_j) from U_j, $\mathcal{B}$ responds $\mathcal{A}$ with ciphertext c_j.
- Challenge. $\mathcal{A}$ selects a challenged set S where $|S| - |S'| > 0$ and two sets of messages $M_0 = \{m_{0j} : U_j \in U\}$ and $M_1 = \{m_{1j} : U_j \in U\}$, and sends them to $\mathcal{B}$. $\mathcal{B}$ flips a coin $b \in \{0, 1\}$ to select message set M_b to be encrypted and aggregated. $\mathcal{B}$ then encrypts m_{bj} with encryption key $a_j = PRF(\alpha_j, r_j)$ where r_j is the randomness for $U_j \in U$, and aggregates all ciphertexts to obtain the aggregated ciphertext c_b. Finally, $\mathcal{B}$ sends $(c_b, \{r_j\}_{U_j \in U})$ to $\mathcal{A}$.
- Guess. $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$. If $b' = b$, we say that $\mathcal{A}$ wins this secure game.

Definition 3. *For any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, our RLDA scheme is semantic secure if the probability of $\mathcal{A}$ winning the above secure game is negligible.*

4 Robust and Lightweight Data Aggregation (RLDA) Scheme

In this section, we first give an overview of the RLDA scheme, and then present the concrete construction.

4.1 Overview

In the RLDA, we wish that the CS can obtain the aggregated data of users without learning individual private data, as well as being robust to the dropped users and ENs. The RLDA consists of system setup phase and data collection phase. In the system setup phase, the CS initiates the data collection system, and sends some necessary system parameters to the ENs and users. The data collection phase is

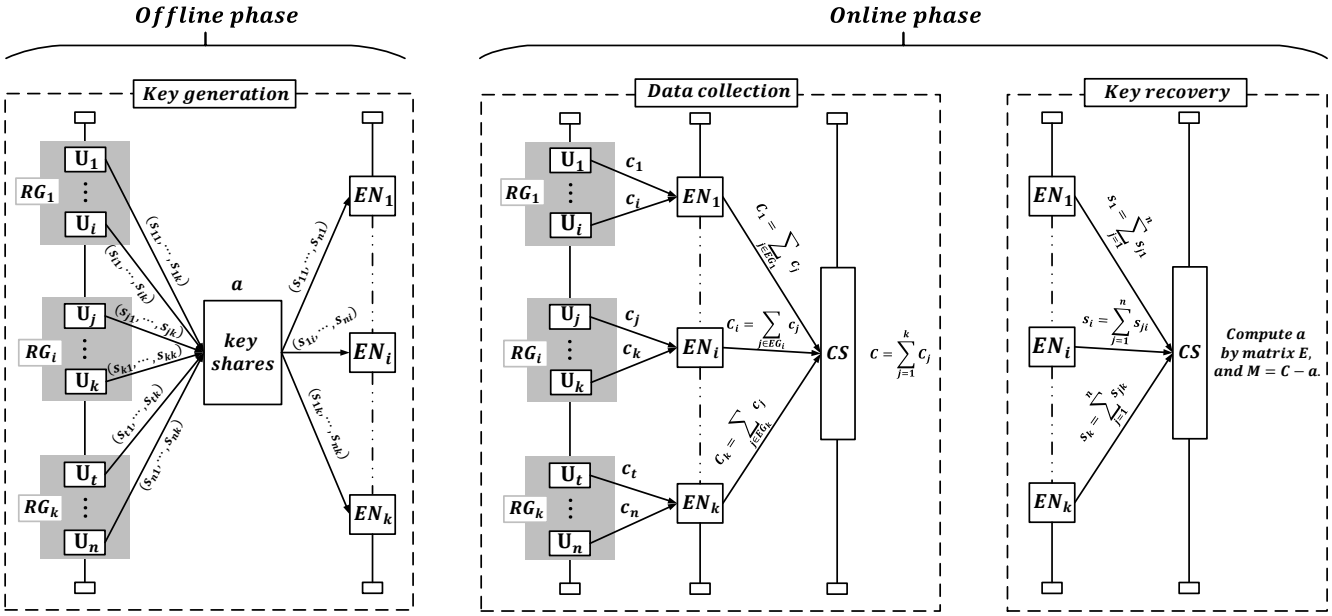


Fig. 3: Work flow of RLDA.

divided into the offline phase and online phase, which is shown in Fig. 3.

During the offline phase, users generate a bunch of encryption keys, and also distribute the key shares to all ENs to make the RLDA perform robustly in the presence of dropout users. Moreover, ENs can verify the well-formedness of key shares to guarantee correct decryption.

In the online phase, users encrypt their data and send to the corresponding ENs, which then aggregate the ciphertexts and send to the CS. To learn the aggregated data, the CS can compute the aggregated masks by requiring ENs to send the sub-masks to the CS. Even in presence of dropped ENs, the CS is still able to robustly and correctly obtain the aggregated results.

4.2 Concrete Construction

In this section, the concrete RLDA scheme is presented.

4.2.1 System Setup Phase

Given a security parameter λ, the CS runs the **Setup**(1^λ) to initialize a data collection system. Specifically, a multiplicative cyclic group $\mathbb{G}$ with order p is selected and a generator g is selected from $\mathbb{G}$, where p is a large prime. A PRF $F : \mathbb{Z}_p^* \times \mathbb{Z}_p^* \to \mathbb{Z}_p^*$ is used to generate encryption keys. The dropout threshold d, collusion threshold t, and recovery threshold e satisfying $k - d > e > t \geq 0$ are selected, where k is the number of ENs. Then, the hyper-invertible matrix[1] $E_{k\times e}$ is constructed with the property that the e-dimension vector $\mathbf{a}$ can be recovered by any e values of encoded vector $(E \times \mathbf{a})_{k\times 1}$. Finally, the system parameter SP is set as $SP = \{p, \mathbb{G}, g, E\}$.

1. A hyper-invertible matrix is a matrix of which every (non-trivial) square submatrix is invertible [34].

4.2.2 Data Collection Phase

In the offline phase, users prepare sufficient encryption keys along with ENs in a verifiable way. In the online phase, users can use the prepared encryption keys to encrypt their own data, and send the ciphertexts to the ENs. At the end of each epoch, ENs aggregate the received ciphertexts and send them to CS. The CS can decrypt the aggregated ciphertexts with the assistance of ENs.

Offline phase. In this phase, users generate the encryption keys and verifiably share them to all ENs, which is illustrated as follows. In the proposed scheme, we use the Vandermonde matrix as the hyper-invertible matrix (Equation (1)) [34, 35], assuming that each EN_i has a unique value $\beta_i \in \mathbb{Z}_p^*$ $(i \in [1, k])$.

$$E_{k\times e} = \begin{bmatrix} 1 & \beta_1 & \cdots & \beta_1^{e-1} \\ 1 & \beta_2 & \cdots & \beta_2^{e-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \beta_k & \cdots & \beta_k^{e-1} \end{bmatrix} \tag{1}$$

- For $i \in [1, n]$, each U_i randomly selects $\alpha_i \in \mathbb{Z}_p^*$ as the key of PRF F. Then, U_i computes the encryption key $a_i = F(\alpha_i, r_i)$, where r_i is randomly selected from $\mathbb{Z}_p^*$. U_i also additively shares the key a_i, i.e.,

  $$a_i = a_{i1} + \cdots + a_{ie},$$

  where $a_{i1}, \cdots, a_{ie} \in (\mathbb{Z}_p^*)^e$, and let $\mathbf{a}_i = (a_{i1}, \cdots, a_{ie})$. Next, U_i computes the commitment $cm_{ij} = g^{a_{ij}}$, and the key share $s_{ij} = E[j] \times \mathbf{a}_i^T$ $(i \in [1, k])$. Finally, U_i securely sends the key share s_{ij} to EN_j, and broadcasts commitment set $\Delta_i = \{cm_{i1}, \cdots, cm_{ie}\}$ to all ENs.
- After receiving Δ_i and key share s_{ij} from U_i, EN_j can check the correctness of key share as follows,

$$g^{s_{ij}} = g^{E[j]\times \mathbf{a}_i^T} = g^{\sum_{l=1}^{e} E[j][l]\cdot a_{ie}} = \prod_{l=1}^{e} cm_{il}^{E[jl]}. \tag{2}$$

The EN_j secretly stores s_{ij} if the verification passes, and aborts otherwise. Note that if any one of ENs aborts in this step, then all ENs will abort.

The verification check (Equation (2)) helps ENs to check that the key shares are indeed encoded by the hyper-invertible matrix E, which guarantees the correct decryption. In the offline phase, users can prepare sufficient encryption keys for secure data collection in the online phase. Thus, it is inefficient to check the key shares individually. In the following, batch check is supported to improve the verification efficiency of ENs.

Batch check. Each U_i prepares w keys $a_i^{(1)}, \cdots, a_i^{(w)}$, and the corresponding commitment sets $\Delta_i^{(1)}, \cdots, \Delta_i^{(w)}$ $(i \in [1, n])$. The EN_j can check the correctness of all nw key shares $\{s_{ij}^{(1)}, \cdots, s_{ij}^{(w)}\}_{i=1}^n$ by Equation (3).

$$\begin{aligned} g^{\sum_{h=1}^{w}\sum_{i=1}^{n} s_{ij}^{(h)}} &= g^{\sum_{h=1}^{w}\sum_{i=1}^{n} E[j]\times \mathbf{a}_i^{(h)T}} \\ &= g^{\sum_{h=1}^{w}\sum_{i=1}^{n}\sum_{l=1}^{e} E[j][l]\cdot a_{il}^{(h)}} \\ &= g^{\sum_{l=1}^{e}(\sum_{h=1}^{w}\sum_{i=1}^{n} \times a_{il}^{(h)})\cdot E[j][l]} \\ &= \prod_{l=1}^{e}(\prod_{h=1}^{w}\prod_{i=1}^{n} cm_{il}^{(h)})^{E[jl]} \end{aligned} \tag{3}$$

Online phase. In this phase, the data collection is divided into several epoches (e.g., every 15 minutes is an epoch). During each epoch, each U_i encrypts its data $m_i \in Z_p^*$, and then sends the ciphertext c_i to the corresponding EN_j. At the ending of the epoch, EN_j aggregates its received ciphertexts $C_j = \sum_{i\in\mathcal{U}_j} c_i$, and sends $(C_j, \mathcal{U}_j)$ to the CS, where $\mathcal{U}_j$ is the surviving set meaning that users in the $\mathcal{U}_j$ have sent the data to the EN_j.

- **Encryption**. The user U_i encrypts data m_i as $c_i = a_i + m_i$, and sends it to EN_j if $i \in RG_j$.
- **Collection**. At the end of the epoch, EN_j first identifies the set of surviving users $\mathcal{U}_j$. When obtaining ciphertexts $\{c_i\}_{i\in\mathcal{U}_j}$ from users in its region, EN_j computes $C_j = \sum_{i\in\mathcal{U}_j} c_i$, and sends $(C_j, \mathcal{U}_j)$ to the CS.
- **Read**. In this step, the CS first ascertains all surviving users, and then requires ENs to compute sub-masks of all surviving users $\mathcal{U}$. If some of ENs fail to transfer the sub-masks, the CS can use the hyper-invertible matrix E to recover the mask by $a = \sum_{i\in\mathcal{U}} a_i$. Finally, the CS can successfully decrypt the ciphertext by the mask a. The concrete steps are described as follows.
  Step 1. The CS first identifies the surviving users $\mathcal{U} = \bigcup_{j=1}^{k} \mathcal{U}_j$, and computes $C = \sum_{j=1}^{k} C_j$.
  Step 2. For $j \in [1, n]$, EN_j is notified to compute the sub-mask $s_j = \sum_{i\in\mathcal{U}} s_{ij}$, and securely sends s_j to the CS.
  Step 3. In **Step 2**, some ENs may drop out, thus the CS can only expect to receive sub-masks of part of ENs. As long as receiving at least e sub-masks, the CS can recover the mask. When receiving the first e sub-masks from ENs (denoted as $\mathcal{E} = \{j_1, \cdots, j_e\}$), the CS can compute

$$[\sum_{i\in\mathcal{U}} a_{i1}, \cdots, \sum_{i\in\mathcal{U}} a_{ie}] = E_{\mathcal{E}}^{-1} \times [s_{j_1}, \cdots, s_{j_e}].$$

  Then, the CS can generate the mask $a = \sum_{j=1}^{e}\sum_{i\in\mathcal{U}} a_{ij}$, and decrypt the ciphertext C by $M = C - a$.

5 Secure Histogram Estimation

In the secure histogram estimation, the CS has a small set $\mathcal{D} = \{0, \cdots, B-1\}$, which is known to the users. For each string, the CS wants to know the number of users who hold the string without learning anything else about user's string.
Technique idea. Given a value $m \in \mathcal{D}$, m is encoded as a B-dimension vector $(v_0, \cdots, v_{B-1})$ where $v_i = 1$ if $i = m$ and $v_i = 0$ otherwise. Then, the vector is compressed into a single value while retaining the homomorphic property of summing values in each dimension. Finally, the CS can collect the sum of v_is by the RLDA scheme in Section 4. After the data collection, the CS can recover the aggregate vector from a single aggregated value. We propose RecAgg algorithms to compress and recover the vector, which are presented in Section 5.2. In the following, we first present the construction of secure histogram estimation scheme, which integrates the RecAgg algorithms with the RLDA to achieve secure histogram estimation. Then, the construction of RecAgg algorithms is proposed.

5.1 Construction of Secure Histogram Estimation

Based on RLDA scheme and RecAgg algorithms, we propose an efficient and secure histogram estimation scheme. In the secure histogram estimation scheme, the aux information (R or $\{b_i\}_{i=1}^n$) in the RecAgg, which satisfies that $R > n$ or $b_i > n$ $(i \in [1, n])$, is pre-selected and known to all ENs.

1) U is associated data $m \in \mathcal{D}$ $(|\mathcal{D}| = B)$, and encodes m as a B-dimension vector $(v_0, \cdots, v_{B-1})$ where $v_i = 1$ if $i = m$ and $v_i = 0$ otherwise.
2) Given aux information and vector $(v_0, \cdots, v_{B-1})$, U executes Encode algorithm (either Encode in the Poly_RecAgg or CRT_RecAgg) to obtain the aggregated value m'.
3) With input m', U proceeds the RLDA scheme to complete the data aggregation.
4) When completing the RLDA scheme, the CS could obtain an aggregated data M. By the Decode algorithm of Poly_RecAgg or CRT_RecAgg, the CS can recover the vector $V = (V_0, \cdots, V_{B-1})$ where V_i is the sum of v_i of all n EN, for every $0 \le i \le B-1$.

5.2 RecAgg Algorithm

We propose two RecAgg algorithms based on algebra operations: polynomial based RecAgg called Poly_RecAgg and Chinese Remainder Theorem (CRT) based RecAgg called CRT_RecAgg, is shown as follows.
Poly_RecAgg. Given a n dimension vector $\mathbf{m} = (m_1, \cdots, m_n) \in \mathcal{D}^n$ where each m_i is a positive integer, $\mathbf{m}$ is regarded as the coefficient vector of a polynomial $f(x)$,

Algorithm 1 Poly_RecAgg

Encode$(\mathbf{m}) \rightarrow (m, aux)$ ▷ aggregation

1: Parse $\mathbf{m} = (m_1, \cdots, m_n)$
2: Select a large integer R, where $R \gg Max\{m_1, \cdots, m_n\}$
3: $aux \leftarrow_n R$
4: $m \leftarrow \sum_{i=1}^{n} m_i R^i$
5: **return** (m, aux)

Decode$(m) \rightarrow (\mathbf{m})$ ▷ recovery

1: Compute $T_0 \leftarrow \frac{m}{R}$
2: **for** $i = 1$ to n **do**
3: $\quad m_i \leftarrow T_{i-1} \bmod R$
4: $\quad T_i \leftarrow \frac{(T_{i-1} - m_i)}{R}$
5: **end for**
6: **return** $\mathbf{m} = \{m_i\}_{i=1}^{n}$

where the constant term $m_0 = 0$. Thus, one can easily aggregate the vector $\mathbf{m}$ into a positive number $f(R)$ by evaluating $f(x)$ at the point R. To recover $\mathbf{m}$, one can successfully obtain $\mathbf{m}$ from $f(R)$ by the pre-designed parameter when the evaluated point R satisfies $R > Max\{m_1, \cdots, m_n\}$. Therefore, given R and $f(R)$, one can compute $\mathbf{m}$ from $f(R)$ by division and modular operations. For example, if $R = 11$, and $f(R) = 1028478$, one can compute $\mathbf{m} = (9, 7, 2, 4, 6)$ by $d_i = T_{i-1}\ mod\ R$ where $T_i = \frac{T_{i-1} - d_i}{R}$ and $T_0 = \frac{f(R)}{R}$ $(i = 1, \cdots, 5)$. The Poly_RecAgg is shown in Algorithm 1.

CRT_RecAgg. The CRT_RecAgg algorithm is based on the Chinese Reminder theorem, and can aggregate the n dimension vector to a single value along with some auxiliary information. Specifically, for a vector $\mathbf{m} = (m_1, \cdots, m_n) \in \mathcal{D}^n$, one can construct an equation system (4) by selecting n co-prime number $(b_1, \cdots, b_n)$. According to the Chinese Reminder theorem, equation system (4) has a unique solution $m = \sum_{i=1}^{n} m_i q_i S_i \bmod S$, where $S = \prod_{i=1}^{n} b_i$, $S_i = \frac{S}{b_i}$ and $q_i = S_i^{-1} \bmod b_i$. Besides, $v_i < b_i$ $(i = 1, \cdots, n)$ is restricted to avoid overflow error. In this way, $\mathbf{m}$ can be represented by m and some auxiliary information $(b_1, \cdots, b_n)$. To obtain $\mathbf{m}$, one only needs to compute n modular operations, i.e., $m_i = m \bmod b_i$ $(i = 1, \cdots, n)$. The CRT_RecAgg is shown in Algorithm 2.

$$\begin{cases} m \equiv m_1 (\bmod\ b_1) \\ m \equiv m_2 (\bmod\ b_2) \\ \quad\quad \vdots \\ m \equiv m_n (\bmod\ b_n) \end{cases} \tag{4}$$

Note that the aggregation is the affine operation on a set of data, and thus the additively homomorphic property is satisfied given the same auxiliary information aux.

Additive homomorphism: For two aggregated data $(m^{(1)}, aux^{(1)})$ and $(m^{(2)}, aux^{(2)})$ of two set of data $(m_1^{(1)}, \cdots, m_n^{(1)})$ and $(m_1^{(2)}, \cdots, m_n^{(2)})$ where $aux^{(1)} = aux^{(2)}$, anyone can create an aggregated data of $(m_1^{(1)} + m_1^{(2)}, \cdots, m_n^{(1)} + m_n^{(2)})$ by computing $m^{(1)} + m^{(2)}$.

5.3 Toward Multidimensional Data Aggregation

In practice, heterogeneous and diverse data will be produced, thus it is essential to support multidimensional data aggregation. In the following, we show that the proposed secure histogram estimation scheme can also collect multidimensional data with carefully designed aux.

Algorithm 2 CRT_RecAgg

Encode$(\mathbf{m}) \rightarrow (m, aux)$ ▷ aggregation

1: Parse $\mathbf{m} = (m_1, \cdots, m_n)$
2: Select n co-prime numbers $\{b_1, \cdots, b_n\}$, where $b_j > d_j$ $(j = 1, \cdots, n)$
3: $aux \leftarrow \{b_1, \cdots, b_n\}$
4: $S \leftarrow \prod_{j=1}^{n} b_i$
5: **for** $j = 1$ to n **do**
6: $\quad S_j \leftarrow \frac{S}{b_j}$
7: $\quad q_j \leftarrow S_j^{-1} \bmod b_j$
8: **end for**
9: $m \leftarrow \sum_{j=1}^{m} m_j q_j S_j \bmod S$
10: **return** (m, aux)

Decode$(m, aux) \rightarrow (\mathbf{m})$ ▷ recovery

1: Parse $aux = \{s_1, \cdots, s_n\}$
2: **for** $j = 1$ to n **do**
3: $\quad m_j \leftarrow m \bmod s_i$
4: **end for**
5: **return** $\mathbf{m} = \{m_i\}_{i=1}^{n}$

By carefully choice of aux, the secure histogram estimation scheme can collect multidimensional data. Assuming that each EN_i has a n-dimension data $\mathbf{m}_i = (m_{i1}, \cdots, m_{in})$, and the aux can be set as:

$$R > Max\{(m_{i1}, \cdots, m_{in})_{i=1}^{n}\},$$

or

$$b_j > \sum_{i=1}^{n} m_{ij}, j \in [1, n].$$

Then, the CS can obtain the aggregated data of each dimension by the secure histogram estimation scheme. At the deployment, the aux can be set larger according to the history data to avoid data overflow.

6 Security Analysis

In this section, we prove the security of RLDA scheme. In the secure histogram estimation scheme, the vector is first encoded to a single value, and then the value is collected by the RLDA scheme. The major difference between RLDA scheme and secure histogram estimation scheme is the phase of local raw data processing, and this phase does not induce the secure attacks in our threat model. Thus, only the security of RLDA scheme is proved in this section.

Theorem 1. *The RLDA scheme is semantically secure against any collusion of at most $(n-1)$ users under the indistinguishability of PRF.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that can break the semantic security of the RLDA scheme with a non-negligible advantage, i.e., $Pr_{\mathcal{A}}^{RLDA}[succ] \geq \varepsilon$. Then, we can construct a challenger $\mathcal{B}$ who can break the indistinguishability of PRF. In the following, $\mathcal{B}$ will use $\mathcal{A}$ as a sub-routine to break the indistinguishability of PRF.

Setup. The adversary $\mathcal{A}$ randomly selects $n-1$ users as the corrupted nodes, of which the PRF keys are known to $\mathcal{A}$. Assume that U_n is the uncorrupted user, and $\{\alpha_i\}_{i=1}^n$ are the PRF keys of all n users.

Query. Given an encryption query (j, m_j, r_j) from $\mathcal{A}$, $\mathcal{B}$ responds $\mathcal{A}$ with $c_j = F(\alpha_i, r_j) + m_j$.

Challenge. The adversary $\mathcal{A}$ selects two sets of messages $M_0 = \{m_{01}, \cdots, m_{0n}\}$ and $M_1 = \{m_{11}, \cdots, m_{1n}\}$, and sends to $\mathcal{B}$. $\mathcal{B}$ flips a coin $b \in \{0,1\}$, and computes $t_b = f(\alpha_n, r_n)$ where r_n is randomly selected from $\mathbb{Z}_p^*$ if $b = 0$. Otherwise, $\mathcal{B}$ randomly selects $t_b \in \mathbb{Z}_p^*$. Then, $\mathcal{B}$ flips a coin $w \in \{0,1\}$, and responds c_w to $\mathcal{A}$, where $c_w = Y_w + t_b$ and $Y_b = \sum_{i=1}^{n} m_{wi} + \sum_{i=1}^{n-1} f(\alpha_i, r_i)$.

Guess. The adversary $\mathcal{A}$ outputs a bit b'. If $b' = w$, $\mathcal{B}$ returns $b'' = 0$ and $b'' = 1$ otherwise.

Let $\mathcal{B}(c_w)$ denote the output of $\mathcal{B}$ on input c_w, the probability of $\mathcal{B}$ breaking the indistinguishability of PRF is

$$\begin{aligned}
&Pr_{\mathcal{B}}^{PRF}[succ] = Pr[b'' = b] \\
&= \frac{1}{2}(Pr[b'' = 0|b = 0] + Pr[b'' = 1|b = 1]) \\
&= \frac{1}{4}(Pr[b'' = 0|b = 0, w = 0] + Pr[b'' = 0|b = 0, w = 1] + \\
&Pr[b'' = 1|b = 1, w = 0] + Pr[b'' = 1|b = 1, w = 1]) \\
&= \frac{1}{4}(Pr[\mathcal{B}(t_0 + Y_0) = 0] + Pr[\mathcal{B}(t_0 + Y_1) = 1] + \\
&Pr[\mathcal{B}(t_1 + Y_0) = 0] + Pr[\mathcal{B}(t_1 + Y_1) = 1]) \\
&= \frac{1}{4}(Pr[\mathcal{B}(t_0 + Y_0) = 0] + Pr[\mathcal{B}(t_0 + Y_1) = 1] + \\
&1 - Pr[\mathcal{B}(t_1 + Y_0) = 0] + Pr[\mathcal{B}(t_1 + Y_1) = 1]).
\end{aligned} \tag{5}$$

The probability of $\mathcal{A}$ to break the semantic security of the RLDA scheme is

$$Pr_{\mathcal{A}}^{RLDA}[succ] = \frac{1}{2}Pr[\mathcal{B}(t_0+Y_0) = 0] + \frac{1}{2}Pr[\mathcal{B}(t_0+Y_1) = 1],$$

where $t_0 + Y_0$ and $t_0 + Y_1$ are the valid ciphertexts.

Since t_1 is randomly selected from $\mathbb{Z}_p^*$, the distribution of $\{t_1 + Y_0\}$ and $\{t_1 + Y_1\}$ is identical. Thus, the probability of $\mathcal{A}$ to distinguish $\{t_1 + Y_0\}$ and $\{t_1 + Y_1\}$ is same, i.e., $Pr[\mathcal{B}(t_1 + Y_0) = 0] = Pr[\mathcal{B}(t_1 + Y_1) = 1]$.

Based on the above two observations, we have

$$Pr_{\mathcal{B}}^{PRF}[succ] = \frac{1}{2}Pr_{\mathcal{A}}^{RLDA}[succ] + \frac{1}{4} \geq \varepsilon + \frac{1}{4}.$$

The challenger can break the indistinguishability of PRF with a non-negligible probability $\varepsilon + \frac{1}{4}$, which is a contradiction to indistinguishability of PRF (Definition 2). Therefore, the RLDA scheme is semantically secure against the collusion of at most $(n-1)$ ENs under the indistinguishability of PRF. □

Theorem 2. *For any pair of dropout threshold d and collusion threshold t satisfying $d + t < k$, our RLDA scheme can simultaneously achieve robust aggregation against up to any d dropped ENs and protect privacy against up to any t colluding ENs.*

Proof. **Robust guarantee**. In the RLDA scheme, all users compute key shares by the hyper-invertible matrix E, and send to ENs for key recovery during the read step of the online phase. In the offline phase, each U_i computes key shares $s_{ij} = E[j] \times \mathbf{a}_i^T$ for EN_j. EN_j will aggregate key shares of all surviving users to obtain sub-mask $s_j = \sum_{i \in \mathcal{U}} s_{ij}$. In presence of d $(k - d > e)$ dropped ENs, the CS can compute masks by the hyper-invertible matrix E:

$$[\sum_{i \in \mathcal{U}} a_{i1}, \cdots, \sum_{i \in \mathcal{U}} a_{ie}] = E_{\mathcal{E}}^{-1} \times [s_{j_1}, \cdots, s_{j_e}].$$

Thus, the CS can recover the aggregated data for the surviving users $\mathcal{U}$ by $\sum_{i \in \mathcal{U}} m_i = C - \sum_{j=1}^{e} \sum_{i \in \mathcal{U}} a_{ij}$.

Privacy guarantee. The privacy guarantee should be satisfied even if the messages sent from all surviving and dropped users and ENs and the message of any set of at most t colluded ENs are received by the CS. Specifically, the privacy guarantee is that the CS cannot infer any additional information about $\{m_i\}_{i \in [1,n]}$ beyond that contained in $\sum_{i \in \mathcal{U}} m_i$ and known from the colluding ENs. That is, for any the set of surviving users $\mathcal{U}$, the set of surviving ENs $\mathcal{U}_{EN}$ and the set of colluding ENs $\mathcal{T}$,

$$I(\{m_i\}_{i \in \mathcal{U}}; \{c_i\}_{i \in \mathcal{U}}, \{s_j\}_{j \in \mathcal{U}_{EN}} | \sum_{i \in \mathcal{U}} m_i, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}}) = 0,$$

where $\mathcal{U} \subset [1, n]$, $\mathcal{U}_{EN} \subset [1, k]$, $|\mathcal{T}| \leq t$, $I(X;Y)$ denotes the mutual information, and $H(Y)$ denotes information entropy.

$$I(\{m_i\}_{i \in \mathcal{U}}; \{c_i\}_{i \in \mathcal{U}}, \{s_j\}_{j \in \mathcal{U}_{EN}} | \sum_{i \in \mathcal{U}} m_i, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}}) \tag{6}$$

$$= H(\{c_i\}_{i \in \mathcal{U}}, \{s_j\}_{j \in \mathcal{U}_{EN}} | \sum_{i \in \mathcal{U}} m_i, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}}) \tag{7}$$

$$- H(\{c_i\}_{i \in \mathcal{U}}, \{s_j\}_{j \in \mathcal{U}_{EN}} | \{m_i\}_{i \in \mathcal{U}}, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}})$$

$$= H(\{c_i\}_{i \in \mathcal{U}}, \{\sum_{i \in \mathcal{U}} s_{ij}\}_{j \in \mathcal{U}_{EN}} | \sum_{i \in \mathcal{U}} m_i, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}}) \tag{8}$$

$$- H(\{c_i\}_{i \in \mathcal{U}}, \{\sum_{i \in \mathcal{U}} s_{ij}\}_{j \in \mathcal{U}_{EN}} | \{m_i\}_{i \in \mathcal{U}}, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}})$$

$$= H(\{c_i\}_{i \in \mathcal{U}} | \sum_{i \in \mathcal{U}} m_i, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}}) \tag{9}$$

$$+ H(\{\sum_{i \in \mathcal{U}} s_{ij}\}_{j \in \mathcal{U}_{EN}} | \sum_{i \in \mathcal{U}} m_i, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}})$$

$$- H(\{c_i\}_{i \in \mathcal{U}} | \{m_i\}_{i \in \mathcal{U}}, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}})$$

$$- H(\{\sum_{i \in \mathcal{U}} s_{ij}\}_{j \in \mathcal{U}_{EN}} | \{m_i\}_{i \in \mathcal{U}}, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}})$$

$$= H(\{c_i\}_{i \in \mathcal{U}} | \sum_{i \in \mathcal{U}} m_i, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}}) \tag{10}$$

$$+ H(\{\sum_{i \in \mathcal{U}} s_{ij}\}_{j \in \mathcal{U}_{EN} \setminus \mathcal{T}} | \sum_{i \in \mathcal{U}} m_i)$$

$$- H(\{c_i\}_{i \in \mathcal{U}} | \{m_i\}_{i \in \mathcal{U}}, \{s_{ij}\}_{i \in \mathcal{U}, j \in \mathcal{T}})$$

$$- H(\{\sum_{i \in \mathcal{U}} s_{ij}\}_{j \in \mathcal{U}_{EN} \setminus \mathcal{T}} | \{m_i\}_{i \in \mathcal{U}})$$

$$= 0. \tag{11}$$

where Equation (8) follows that $\{s_j\}_{j \in \mathcal{U}_{EN}}$ is invertible to $\{\sum_{i \in \mathcal{U}} s_{ij}\}_{j \in \mathcal{U}_{EN}}$. Equation (9) follows from the chain rules. In Equation (10), the second and the last terms follow from the independence of s_{ij}'s and m_i's. Equation (11) follows from that 1) $\{\sum_{i \in \mathcal{U}} s_{ij}\}_{j \in \mathcal{U}_{EN} \setminus \mathcal{T}}$ is a function of $\sum_{i \in \mathcal{U}} m_i$ and

TABLE 2: Functionality comparison

Functionality	SecAgg[18]	VeriFL[19]	LightSecAgg[22]	CodedSecAgg[23]	TurboAgg[21]	SwiftAgg[24]	InfoAgg[25]	RLDA
Verifiability	✗	✓	✗	✗	✗	✗	✗	✓
Robust guauantee	✓	✓	✓	✓	✓	✓	✓	✓
Privacy guarantee	✓	✓	✓	✓	✓	✓	✓	✓
Multidimensional aggregation	✗	✗	✗	✗	✗	✗	✗	✓
Histogram estimation	✗	✗	✗	✗	✗	✗	✗	✓
User independence	✗	✗	✗	✗	✗	✗	✗	✓
Online/offline paradigm	✗	✗	✗	✗	✗	✗	✗	✓

TABLE 3: Complexity comparison in the online phase

Schemes	U			EN			CS		
	TurboAgg	SwiftAgg	RLDA	TurboAgg	SwiftAgg	RLDA	TurboAgg	SwiftAgg	RLDA
online comm.	$O(\frac{n}{k})$	$O(\frac{n}{k})$	$O(1)$	–	–	$O(1)$	$O(1)$	$O(1)$	$O(1)$
online comp.	$O(\frac{n^2}{k})$	$O(\frac{n}{k}t_n^2+n)$	$O(1)$	–	–	$O(\frac{n}{k})$	$O(1)$	$O(t_n^2)$	$O(e^2)$

$\{s_{ij}\}_{i\in\mathcal{U},j\in\mathcal{T}}$; 2) s_{ij}'s are independent from m_i's; 3) $\{c_i\}_{i\in\mathcal{U}}$ is a function of $\{m_i\}_{i\in\mathcal{U}}$ and $\{s_{ij}\}_{i\in\mathcal{U},j\in\mathcal{T}}$ combined with the non-negativity of mutual information. □

7 PERFORMANCE EVALUATION

In this section, we first conduct the functionality comparison between our RLDA and existing schemes based on masked encryption. Then the complexity of RLDA is analyzed, and simulation experiments are conducted to evaluate performance.

7.1 Functionality comparison

As shown in Table 2, we comprehensively compare the functionalities of schemes [18, 19, 21–25], which are masked encryption data aggregation schemes, from the aspects of data security (e.g., verifiability, robust guarantee and privacy guarantee), data analysis functionality (e.g., multidimensional aggregation, histogram estimation), and user independence. RLDA provides data security properties as proved in Section 6. Based on the proposed RecAgg algorithms, RLDA supports histogram estimation and multidimensional aggregation. For constrained users, user independence means that each user only needs to conduct its own work, and does not need to assist the key generation, data transferring, key recovery, etc. In the RLDA, users only need to submit their own data to the ENs, while in existing schemes [18, 19, 21–25], the user has to assist other users to transfer data. Thus, our RLDA scheme provides user independence, which is vital for resource-constrained IoT users.

7.2 Complexity analysis

In this section, we evaluate the complexity of TurboAgg [21], SwiftAgg [24], and our RLDA, where TurboAgg and SwiftAgg support group-based aggregation and also provide robustness in presence of dropped ENs. The storage cost, communication cost and computation cost are measured in units of elements $|\mathbb{Z}_p|$ and $|\mathbb{G}|$, or operation in $\mathbb{Z}_p^*$ and $\mathbb{G}$, which are shown in Table 3 and Table 4.

Offline storage cost. Each U_i independently generates an encryption key a_i for online encryption, which incurs constant storage cost $|\mathbb{Z}_p|$. Additionally, each U_i ($i \in [n]$) computes key shares s_{ij} for EN_j ($j \in [k]$), allowing key recovery in presence of dropped ENs. Thus, the total storage cost at each EN is $n|\mathbb{Z}_p|$.

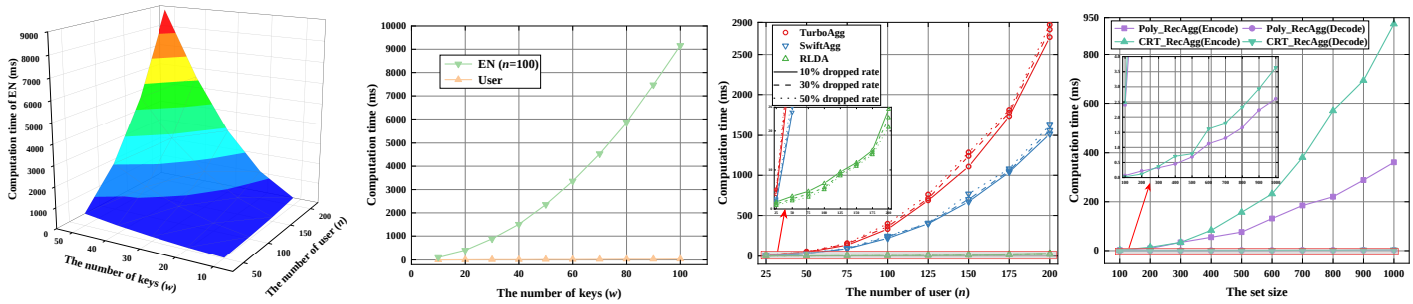
TABLE 4: Complexity of RLDA in the offline phase

Schemes	U	EN
offline comm.	$O(k+e)$	–
offline comp.	$O(ke)$	$O(n+e)$
offline storage	$O(1)$	$O(n)$

Offline communication and computation cost. During the offline phase, each user prepares encryption keys and distributes key shares to ENs. Specifically, each U_i computes key shares s_{ij} by a hyper-invertible matrix with dimension $k \times e$ for every EN_j ($j \in [k]$), and also generates commitments Δ_i for share verification. Hence, the offline communication and computation cost of RLDA at each user is $O(k+e)$ and $O(ke)$. The offline computation cost at each EN is $O(n+e)$, which is to verify key shares.

Online communication and computation cost. The major communication cost lies in data encoding for TurboAgg [21], SwiftAgg [24]. In the TurboAgg, each user in i-th group has to encode data by Lagrange coding [31] and shares data with every user in the $i+1$-th group by polynomial evaluation, thus incurring $O(\frac{n}{k})$ communication cost and $O(2(\frac{n}{k})^2)$ computation cost. The SwiftAgg divides aggregation into intra-group aggregation and inter-group aggregation. In the intra-group aggregation, each user shares its data to the remaining users in the current group by secret sharing. Then each user aggregates all shares and sends the aggregated shares to intended user in next group. Hence, the communication cost and computation cost are $O(\frac{n}{k})$ and $O(\frac{n}{k}(t_n)^2+n)$, respectively, where t_n is the colluded threshold. For our RLDA, the user only needs to encrypt data, and has a constant cost. EN is required to aggregate data, which causes $O(\frac{n}{k})$ computation cost.

For overhead at the CS, the major cost is key recovery. For the TurboAgg, key recovery is completed by users in the final group, and incurs $O(k(\frac{n}{k})^2)$ computation cost in the worst case. In the SwiftAgg, the CS can recover data with $O(t_n^2)$ computation cost when receiving data from any



(a) Computation time of ENs for key verification in the offline phase.

(b) Computation time of key generation in the offline phase.

(c) Total running time of RLDA versus TurboAgg and SwiftAgg in the online phase.

(d) Computation time of Poly_RecAgg and CRT_RecAgg.

Fig. 4: Computation time of RLDA and comparison with the existing schemes.

t_n+1 users in the last group. For our RLDA, the CS recovers key by the hyper-invertible matrix, which requires $O(e^2)$ computation cost.

7.3 Experiment results

We evaluate the performance of RLDA and compare it with TurboAgg [21] and SwiftAgg [24]. The experiments were conducted on a desktop with a 2.90 GHz Intel Core i7-10700 CPU and 4 GB of memory. The operating system used was Ubuntu 16.04LTS, and the programming language used was Python with programming done using the GNU Multiple Precision Arithmetic (GMP) Library[2]. Each experiment was repeated 20 times to obtain the average results.

In the offline phase, all users generate keys for online encryption and distribute key shares to ENs for efficient key recovery, which also do not require them to be online after submitting the data. The major computation cost of ENs is to verify the consistence of key shares, which linearly increase with the number of users and the number of keys, as shown in Fig. 4(a). When $n = 200$ and $w = 10$, the time cost of ENs to check shares is 384.51 ms. We also evaluate the offline computation cost of users, as shown in Fig. 4(b). With the increase of keys (w ranges from 10 to 100), the computation cost of each user varies from 3.87 ms to 36.37 ms. Additionally, when fix $n = 100$, the computation time of each EN to verify key shares is from 0.1 s to 9 s. All computation cost is conducted offline to generate sufficient keys for online encryption.

In the online phase, the major cost lies in data collection and key recovery. We evaluate the performance of our proposed RLDA, and TurboAgg and SwiftAgg with different dropout rates. The RLDA provides a privacy guarantee $t = \frac{k}{2}$ since only the collusion of ENs could compromise the privacy of users. Because users take on the relay data and key recovery in TurboAgg and SwiftAgg, the privacy guarantee of these two schemes relies on users' number and is set as $t = \frac{n}{2}$. For the dropped users, qn users or qk ENs are randomly selected to model the dropped users or ENs, where q is the dropout rate. Three different dropout rates are selected, $q = 0.1$, $q = 0.3$, $q = 0.5$, which are realistic values according to the industrial observation in real system [36]. We set the number of ENs $k = 10$, and the number of users varies from 25 to 200 in increments of 25. In the RLDA, as the dropped rate increases, the threshold decreases, which indicates that the number of surviving ENs also decreases. Since key recovery is completed by submasks of surviving ENs, the computation time gradually decreases as the dropout rate increases. While in TurboAgg and SwiftAgg, the keys of dropped users are required to be recovered, and thus the computation time increases as the dropout rate increases. The experiment results are shown in Fig. 4(c). As the RLDA introduces ENs to assist data collection and key recovery, only a few computation time is required to recover keys in assistance with surviving ENs. Thus, the computation time of RLDA is significantly lower than that of TurboAgg and SwiftAgg.

The proposed secure histogram estimation scheme is implemented by RLDA scheme and RecAgg algorithms. The performance of RLDA has been demonstrated in the above analysis. In the following, we mainly evaluate the performance of the two RecAgg algorithms: Poly_RecAgg and CRT_RecAgg. The set size of $\mathcal{S}$ varies from 100 to 1000, and the efficiency of encode and decode algorithms of Poly_RecAgg and CRT_RecAgg is shown in Fig. 4(d). From Fig. 4(d), the computation time of Poly_RecAgg is lower than CRT_RecAgg. When $B = 500$, Poly_RecAgg takes 76.04 ms to encode the vector and decodes the vector with 0.68 ms, while CRT_RecAgg takes 157.03 ms for encoding the vector and requires 0.79 ms to decode the vector.

8 Conclusion

This paper designed a robust and lightweight data aggregation (RLDA) scheme in edge-cloud systems. RLDA uses an offline/online paradigm with ENs for verifiable key generation, data aggregation and key recovery. By recovering the keys of surviving ENs, our RLDA significantly reduced the overhead of communication and computation. We also proposed two recoverable aggregation algorithms to support secure histogram estimation, which achieve the transformation between vector and single value. The simulation experiments show that our RLDA is computationally lightweight, especially in the aspect of key recovery. Our RLDA makes a meaningful attempt to design a practical data aggregation scheme with privacy preservation, which

2. https://gmplib.org/

is resilient for IoT users in edge-cloud systems. In the future work, we will explore the efficient data analysis over encrypted data and database, such as private heavy hitters and hidden query.

REFERENCES

[1] F. Shirin Abkenar, P. Ramezani, S. Iranmanesh, S. Murali, D. Chulerttiyawong, X. Wan, A. Jamalipour, and R. Raad, "A survey on mobility of edge computing networks in iot: State-of-the-art, architectures, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2329–2365, 2022.

[2] Y. Wang, Z. Su, N. Zhang, R. Xing, D. Liu, T. H. Luan, and X. Shen, "A survey on metaverse: Fundamentals, security, and privacy," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 319–352, 2023.

[3] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, and A. P. Sheth, "Machine learning for internet of things data analysis: a survey," *Digital Communications and Networks*, vol. 4, no. 3, pp. 161–175, 2018.

[4] Y. Wang, Y. Pan, M. Yan, Z. Su, and T. H. Luan, "A survey on chatgpt: Ai–generated contents, challenges, and solutions," *IEEE Open Journal of the Computer Society*, vol. 4, pp. 280–302, 2023.

[5] G. Wang, B. Wang, T. Wang, A. Nika, H. Zheng, and B. Y. Zhao, "Defending against sybil devices in crowdsourced mapping services," in *Proceedings of the 14th annual international conference on mobile systems, applications, and services (MobiSys'16)*, 2016, pp. 179–191.

[6] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, robust, and scalable computation of aggregate statistics," in *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI'17)*, 2017, pp. 259–282.

[7] J.-N. Liu, J. Weng, A. Yang, Y. Chen, and X. Lin, "Enabling efficient and privacy-preserving aggregation communication and function query for fog computing-based smart grid," *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 247–257, 2020.

[8] Y. Zhan, L. Zhou, B. Wang, P. Duan, and B. Zhang, "Efficient function queryable and privacy preserving data aggregation scheme in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 3430–3441, 2022.

[9] L. Zhao, J. Jiang, B. Feng, Q. Wang, C. Shen, and Q. Li, "Sear: Secure and efficient aggregation for byzantine-robust federated learning," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3329–3342, 2022.

[10] X. Zhang, C. Huang, Y. Zhang, and S. Cao, "Enabling verifiable privacy-preserving multi-type data aggregation in smart grids," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 6, pp. 4225–4239, 2022.

[11] R. Zhu, M. Li, J. Yin, L. Sun, and H. Liu, "Enhanced federated learning for edge data security in intelligent transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2023.

[12] J. Zhang and C. Dong, "Secure and lightweight data aggregation scheme for anonymous multi-receivers in wban," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 1, pp. 81–91, 2023.

[13] Y. Zhao and J. Chen, "A survey on differential privacy for unstructured data content," *ACM Computing Surveys*, vol. 54, no. 10s, 2022.

[14] E. Shi, H. Chan, E. Rieffel, R. Chow, and D. Song, "Privacy-preserving aggregation of time-series data," in *Annual Network & Distributed System Security Symposium (NDSS'11)*, 2011, pp. 1–17.

[15] T. H. H. Chan, E. Shi, and D. Song, "Privacy-preserving stream aggregation with fault tolerance," in *Financial Cryptography and Data Security: 16th International Conference (FC'2012)*, 2012, pp. 200–214.

[16] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[17] B. Jia, X. Zhang, J. Liu, Y. Zhang, K. Huang, and Y. Liang, "Blockchain-enabled federated learning data protection aggregation scheme with differential privacy and homomorphic encryption in iiot," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 6, pp. 4049–4058, 2022.

[18] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS'17)*, 2017, pp. 1175–1191.

[19] X. Guo, Z. Liu, J. Li, J. Gao, B. Hou, C. Dong, and T. Baker, "Verifl: Communication-efficient and fast verifiable aggregation for federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1736–1751, 2021.

[20] J. Song, W. Wang, T. R. Gadekallu, J. Cao, and Y. Liu, "Eppda: An efficient privacy-preserving data aggregation federated learning scheme," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 5, pp. 3047–3057, 2023.

[21] J. So, B. Güler, and A. S. Avestimehr, "Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 1, pp. 479–489, 2021.

[22] J. So, C. He, C.-S. Yang, S. Li, Q. Yu, R. E. Ali, B. Guler, and S. Avestimehr, "Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning," in *Proceedings of Machine Learning and Systems (MLSys'22)*, vol. 4, 2022, pp. 694–720.

[23] R. Schlegel, S. Kumar, E. Rosnes, and A. G. i. Amat, "Codedpaddedfl and codedsecagg: Straggler mitigation and secure aggregation in federated learning," *IEEE Transactions on Communications*, vol. 71, no. 4, pp. 2013–2027, 2023.

[24] T. Jahani-Nezhad, M. A. Maddah-Ali, S. Li, and G. Caire, "Swiftagg: Communication-efficient and dropout-resistant secure aggregation for federated learning with worst-case security guarantees," in *2022 IEEE International Symposium on Information Theory (ISIT '22)*, 2022, pp. 103–108.

[25] Y. Zhao and H. Sun, "Information theoretic secure

aggregation with user dropouts," *IEEE Transactions on Information Theory*, vol. 68, no. 11, pp. 7471–7484, 2022.

[26] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[27] Y. Nie, W. Yang, L. Huang, X. Xie, Z. Zhao, and S. Wang, "A utility-optimized framework for personalized private histogram estimation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 4, pp. 655–669, 2019.

[28] A. Cheu and M. Zhilyaev, "Differentially private histograms in the shuffle model from fake users," in *2022 IEEE Symposium on Security and Privacy (SP'22)*, 2022, pp. 440–457.

[29] L. Burkhalter, A. Hithnawi, A. Viand, H. Shafagh, and S. Ratnasamy, "TimeCrypt: Encrypted data stream processing at scale with cryptographic access control," in *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI'20)*, 2020, pp. 835–850.

[30] Y. Su, Y. Li, J. Li, and K. Zhang, "Lceda: Lightweight and communication-efficient data aggregation scheme for smart grid," *IEEE Internet of Things Journal*, vol. 8, no. 20, pp. 15 639–15 648, 2021.

[31] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and S. A. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security, and privacy," in *The 22nd International Conference on Artificial Intelligence and Statistics (AISTATS'19)*. PMLR, 2019, pp. 1215–1225.

[32] G. Oded, *Foundations of Cryptography: Volume 2, Basic Applications*, 1st ed. USA: Cambridge University Press, 2009.

[33] O. Goldreich, S. Goldwasser, and S. Micali, "How to construct random functions," *Journal of the ACM*, vol. 33, no. 4, pp. 792–807, 1986.

[34] Z. Beerliová-Trubíniová and M. Hirt, "Perfectly-secure mpc with linear communication complexity," in *Theory of Cryptography Conference (TCC'08)*, 2008, pp. 213–230.

[35] A. Klinger, "The vandermonde matrix," *The American Mathematical Monthly*, vol. 74, no. 5, pp. 571–574, 1967.

[36] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan *et al.*, "Towards federated learning at scale: System design," *Proceedings of machine learning and systems (MLSys'19)*, vol. 1, pp. 374–388, 2019.