mathematics

MDPI

Article

TD-RCRF: A Privacy-Preserving Truth Discovery Resistant to Collusion and Reputation Fraud in Mobile Crowdsensing

Libo Ban [1], Lei Wu [1,*], Wei Wu [1,*] and Haipeng Peng [2]

[1] School of Computer Science and Artificial Intelligence, Shandong Normal University, Jinan 250358, China; banlibo2022@163.com
[2] Information Security Center, State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China; penghaipeng@bupt.edu.cn
* Correspondence: wulei@sdnu.edu.cn (L.W.); wuw@sdnu.edu.cn (W.W.)

Abstract

Privacy-preserving truth discovery (PPTD) has garnered significant attention in mobile crowdsensing (MCS). However, existing research lacks sufficient privacy protection and is often vulnerable to collusion attacks among malicious participants. Moreover, incorrect data submitted by unreliable users and their weights may reduce the accuracy of truth discovery. To address these issues, this paper proposes a privacy-preserving truth discovery framework resistant to collusion and reputation fraud (TD-RCRF) that is highly resistant to collusion and reputation fraud. The scheme employs additive secret sharing to protect sensing data, weights, intermediate results, and ground truth. To screen trustworthy users who meet reputation requirements under the non-colluding dual-server model, we propose a privacy-preserving reputation verification algorithm that combines Pedersen commitment and zero-knowledge proof to verify the validity of mobile users' reputation values. Additionally, we propose a homomorphic strategy that converts shares between multiplication and addition and use it to design a lightweight truth discovery algorithm that further improves the accuracy of the "truth" using reputation values. Security analysis proves that TD-RCRF is privacy-preserving and secure under the non-colluding dual-server assumption. Theoretical analysis and experiments show that it is practical and efficient.

Keywords: truth discovery; secret sharing; zero-knowledge proof; mobile crowdsensing; privacy preservation

MSC: 68M25

Check for updates

Academic Editor: Jonathan Blackledge

Received: 29 March 2026
Revised: 16 April 2026
Accepted: 19 April 2026
Published: 27 April 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The Mobile Cluster Sensing System (MCS) [1] uses innovative terminal technology and mobile communications to analyze data collected by sensors through the cloud to monitor the real world efficiently. The system finds broad applicability in domains like environmental sensing [2] and intelligent transportation services [3]. Nonetheless, MCS faces challenges in real-world applications, including unstable sensor performance, incomplete data collection, background interference, and possible malicious tampering, which can lead to a lack of accuracy in sensor data submitted by individual users.

In response to the described problem, the truth discovery (TD) [4,5] technique has emerged, which has been applied to deal with scenarios containing noisy and conflicting data. Unlike the aforementioned basic method, the truth discovery mechanism adaptively assigns varying importance levels to user contributions according to their data reliability.

This process proceeds iteratively, beginning with an assessment of data credibility for each participant, followed by the computation of a weighted aggregation for each statement as an estimation of the ground truth. As the iterations proceed, the system gradually adjusts the weights of the user data according to how close they are to the true situation so that those users who are closer to the truth receive greater influence. As a result, in subsequent iterations, data provided by highly weighted users are more likely to be recognized as the truth. This process continues until a stable estimate is reached, i.e., the estimated truthfulness converges.

In recent developments, privacy-preserving truth discovery (PPTD) [6–10] has emerged as a key research focus and is centered on efficiently extracting valuable information while ensuring user privacy and security. This field not only covers everyday sensory data but also delves into the realm of highly sensitive private information, such as weight, for example. As a key indicator of the data provider's credibility, data weight contains sensitive information that must be carefully safeguarded since it could be exploited to infer personal attributes such as educational background, capabilities, or even personality characteristics. For example, when discussing complex social issues, although we can gather wisdom from users' opinions, the leakage of weighting information may inadvertently reveal users' intelligence level and educational background. To address the above challenges, Zhang et al. [11] proposed an optimized PPTD architecture that achieves both enhanced computational performance and strong protection of device-level privacy with minimal resource consumption. However, state-of-the-art schemes like this one still face challenges in practical deployment, especially relying on stable and frequent communication between users and cloud servers. In complex and changing network environments, factors such as network instability, human interference, and insufficient device power may cause data transmission to be blocked, thus affecting the smooth operation of the entire system. Therefore, future research needs to further explore how to enhance the robustness and adaptability of the system while safeguarding privacy and efficiency so as to better adapt to the needs of practical application scenarios.

Recently, efforts have been devoted to discovering a series of lightweight strategies aimed at alleviating system overload under resource constraints, especially optimized for user workload. These innovative approaches generally adopt a two-server architecture as their core design, effectively dispersing the heavy burden of user operations under the traditional single-server model to the cloud environment. However, there is still a significant challenge in the existing approaches: the raw or processed truth data is directly exposed to cloud service providers, which may become a high-risk point for privacy leakage and misuse. Therefore, preventing attacks by snooping-motivated cloud service providers is an essential part of protecting user privacy. To overcome these challenges, including overburdened users, uneven distribution of cloud resources, insufficient cloud attack defense capabilities, and dependence on third parties, this paper proposes a privacy-preserving truth discovery framework resistant to collusion and reputation fraud (TD-RCRF) scheme for multi-cloud systems. The core advantage of this scheme is that it can effectively reduce the workload of users and requesters while still providing strong privacy and accuracy guarantees. In addition, we introduce a generalized dual-cloud model that is compatible with various types of truth discovery algorithms and effectively defends against attacks launched by eavesdroppers by tampering with user-to-cloud sensory data transmissions. The main contributions of this paper are as follows:

- We design a privacy-preserving truth discovery framework resistant to collusion and reputation fraud named TD-RCRF. This framework uses additive secret sharing technology to protect sensing data, weights, and true value privacy, and utilizes Pedersen commitment to protect reputation value privacy. Meanwhile, this framework

can effectively screen out high-reputation mobile users and aggregate their data in a collusion-resilient manner, thereby improving the accuracy of the sensing task ground truths.

- We design two protocols to address the challenges posed by conventional secret sharing techniques in division and logarithmic operations. By integrating the additive homomorphic properties of additive secret sharing with the multiplicative homomorphic properties of multiplicative secret sharing, the shares can perform arithmetic operations within the truth-finding algorithm via protocols in a collusion-resistant setting.
- In order to screen out a group of high-reputation users that meet the requirements while resisting collusion and reputation fraud, we propose a reputation verification algorithm. By combining Pedersen commitment and zero-knowledge proof to verify the validity of user reputation values, we design a range verification algorithm to ensure that user reputation values meet the requirements of tasks.
- Our security analysis verified the privacy of TD-RCRF, and experimental evaluations verified the feasibility of the scheme. Compared with other schemes, this scheme not only guarantees privacy and usability but also achieves high computational and communication efficiency.

This paper is structured as follows. Section 2 outlines existing research and its constraints, whereas Section 3 addresses preliminary work. Section 4 then outlines the problem statement, followed by Section 5 detailing the TD-RCRF scheme's design. Section 6 conducts the theoretical analysis, Section 7 conducts the performance evaluation, and Section 8 concludes this paper.

2. Related Work

According to the research and analysis of the related literature, many researchers have developed a variety of truth discovery methods for privacy preservation. However, the plaintext strategies adopted by these methods often fail to perform effectively in real-world mobile crowdsensing scenarios. Variations in device capabilities and economic limitations lead to uneven user engagement, posing challenges for certain schemes to accommodate such dynamic participation environments. To address this issue, Li et al. [12] introduced an improved approach known as the confidence-enhanced truth discovery method. This algorithm can better adapt to scenarios with different user engagement levels and thus produce more accurate truth value results.

In recent years, truthful information extraction techniques, especially truthful value discovery methods, have demonstrated significant advantages in fusing multi-source data to obtain accurate information, attracting extensive attention from both academia and industry. Compared to traditional data aggregation means, such as averaging calculations and voting mechanisms, emerging truth discovery strategies, such as heterogeneous conflict resolution and its optimized version, significantly enhance the credibility of the results by incorporating device reliability considerations. Notably, many of the leading schemes operate without encryption, overlooking the critical need to safeguard participants' privacy throughout data sharing and processing.

Meng et al. [13] pioneered a truth discovery strategy under conflict data parsing, aiming to distill the true situation from contradictory information provided by multiple entities. However, previous truth discovery techniques have mostly focused on the explicit processing category, with insufficient consideration of participants' privacy protection. To overcome this limitation, recent research has increasingly emphasized privacy-preserving truth discovery (PPTD), striving for efficient analysis while maintaining user privacy. Among them, Xu et al. [14] proposed an efficient truth value discovery approach that integrates a carefully planned secure data processing framework [15] with a super-incremental

sequencing method [16,17]. This mechanism enables lightweight aggregation of private data by injecting user-specific random noise and employing a cloud-based error correction technique. However, in the face of the common data disruption problem in mobile crowdsourcing environments, where some users stop reporting data, the scheme faces the challenge of impaired accuracy of aggregation results due to missing noise values that cannot be compensated.

Similarly, Miao et al. [18] contributed the L-PPTD and L2-PPTD protocols; although this is a breakthrough in privacy preservation, the former still requires workers to iteratively compute the weights, and the latter allows workers to be offline but sacrifices the weight privacy in the cloud. Zheng et al. [6] further expanded the scope of privacy-aware truth discovery by designing, respectively, a single-server vs. multi-server environment. To accommodate different deployment models, mechanisms were developed for both centralized and distributed server settings. In the single-server case, fault tolerance was achieved through recalculating blinding factors, though this led to increased communication between participants. The multi-server design offloads part of the computation to servers; however, it still requires user interaction during iterative updates. Drawing inspiration from L2-PPTD, Zheng et al. [19] introduced a secure truth discovery approach that integrates a garbled circuit (GC) and additive homomorphic encryption to minimize user-side involvement. However, the demand for GC generation leads to the rising computational cost of cloud platforms, which becomes a bottleneck for efficiency improvement. In pursuit of more efficient privacy protection schemes, current research trends favor the use of random noise mechanisms.

In the intersecting research fields of reliability assessment and privacy preservation, recent years have shown a technical trend in evolving from a single trust model to a privacy-enhanced framework. Hu et al. [20] made significant progress in the area of fleet service recommendation for in-vehicle networks by proposing a trustworthy scheme based on trust that accurately identifies and recommends reliable fleet headers by calculating the trust value to ensure the high quality of the service. Ni et al. [21] did not take privacy into account, although they utilized reputation thresholds to assess perceived data quality. In response to this challenge, Zhang et al. [11] proposed a privacy-aware fleet recommendation approach tailored for in-vehicle networks, aiming to achieve a trade-off between data utility and privacy protection. To further enhance privacy protection, Liu et al. [22] designed two privacy-protecting trust management approaches. These approaches both evaluate the credibility of sensory data and achieve the finite unlinkability of trust values, thus constructing a robust defense for users' privacy. Cheng et al. [23], on the other hand, focusing on the dual goals of efficiency enhancement and privacy preservation, proposed an in-car network, which is an efficient reputation management lightweight privacy protection scheme for in-vehicle networks, showing good application prospects. At the same time, Ma et al. [24] proposed a pair of reputation control schemes with privacy protection features, specifically designed for edge computing scenarios. These mechanisms safeguard user privacy and efficiently address the issue of malicious participants. Yan et al. [25] similarly leverage reputation metrics to enhance sensing accuracy; however, privacy exposure remains a potential issue. Notably, many of these approaches assume uniform reliability across selected participants, overlooking real-world variations in trustworthiness.

3. Preliminaries

3.1. Truth Discovery

Truth discovery technology can extract reliable and accurate information from multiple sensors. Its principle is that users who frequently provide valid information are given higher weights because their data is closer to the truth. The CRH data framework is

extensively employed as an algorithm across numerous PPTDs. The conventional truth discovery framework comprises three key components: weight update, truth update, and truth convergence.

(1) Weighting Update

An initial base truth is randomly set at this stage, $\{x_m^*\}_{m=1}^M$. User u_k can determine their own weight based on the proximity of their perceived data $x_m^k (m \in \{1, 2, \cdots, M\}$, $K \in \{1, 2, \cdots, K\})$ to the baseline truth. The principle of weight allocation is that the smaller the difference between user u_k's perceived data x_m^k and the baseline truth x_m^*, the closer their data is to the baseline truth, and thus, the user will be assigned a higher weight. In general, the weighting formula for user u_k is shown below:

$$\omega_k = \log(\sum_{k=1}^{K}\sum_{m=1}^{M} d(x_m^k, x_m^*)) - \log(\sum_{m=1}^{M} d(x_m^k, x_m^*)) \tag{1}$$

The distance function $d(\cdot)$ represents the gap between the perceived data x_m^k and the base truth x_m^*.

(2) Truth Update

When the weight w_k of each mobile user u_k is determined, after many iterations, the final truth x_m^* can be calculated based on the perceived data $x_m^k (m \in \{1, 2, \cdots, M\}$, $K \in \{1, 2, \cdots, K\})$. The specific calculation process is as follows:

$$x_m^* = \frac{\sum_{k=1}^{K} w_k \cdot x_m^k}{\sum_{k=1}^{K} w_k} \tag{2}$$

This equation shows that mobile users having higher weights increases the likelihood of their sensing data being regarded as the reference standard.

(3) Truth Convergence

These two steps are repeatedly executed until the benchmark truth satisfies the defined convergence criteria. These criteria may include a fixed number of iterations or a maximum allowable difference between consecutive results, as specified in this paper:

$$\left\| [x_m^*]^{I_m} - [x_m^*]^{I_{m-1}} \right\| < \epsilon \tag{3}$$

Here, $\| \cdot \|$ symbolizes the gap between the baseline truth of two successive iterations, while I_m signifies the iteration count.

3.2. Secret Sharing

Secret sharing is a technique proposed by Shamir for splitting secret data into multiple shared secrets, where data can only be recovered if a certain number of shared secrets are reached. Additive secret sharing (ASS) is defined over a group $\mathbb{F}$ and employs an (n, n) threshold scheme, where the secret $x \in \mathbb{F}$ is randomly partitioned into n shares and no secret can be recovered for any missing share. In this paper, we only consider the case $n = 2$.

Beaver triples are used for safe multiplication operations in ASS and contain a, b, c, satisfying $c = a \cdot b$, where a and b are randomly generated and shared in the offline phase. Before the protocol execution, the data shares and the Beaver triad shares are sent to each server. In the online phase, the server calculates $d_i = x_i - a_i$ and $e_i = y_i - b_i$ and their interactions with each other, ultimately calculating $[z]_i = \frac{d \cdot e}{2} + d \cdot b_i + e \cdot a_i + c_i$ such that $z_1 + z_2 = x \cdot y$. The detailed steps of this algorithm are shown in Algorithm 1.

Algorithm 1 Secure Multiplication Protocol $[\![z]\!] \leftarrow SecMul([\![x]\!], [\![y]\!])$

Input: Shared secret $[\![x]\!]$ and $[\![y]\!]$. Beaver triple: $[\![a]\!]$, $[\![b]\!]$, $[\![c]\!]$, Where $c = a \cdot b$.
Output: Shared secret $[\![z]\!]$ such that $x \cdot y$.
1: S_1 computes $d_1 = x_1 - a_1, e_1 = y_1 - b_1$.
2: S_2 computes $d_2 = x_2 - a_2, e_2 = y_2 - b_2$.
3: S_1 and S_2 exchange the shares $[\![d]\!]$ and $[\![e]\!]$.
4: S_1 computes $z_1 = \frac{d \cdot e}{2} + d \cdot b_1 + e \cdot a_1 + c_1$.
5: S_2 computes $z_2 = \frac{d \cdot e}{2} + d \cdot b_2 + e \cdot a_2 + c_2$.
6: **Return** $[\![z]\!]$.

Multiplicative secret sharing (MSS) is defined over the real number field $\mathbb{R}$. The secret $u \in \mathbb{R}$ is randomly divided into n shares, and the secret cannot be recovered without any of the shares. MSS also uses an (n, n) threshold scheme. It should be noted that MSS is not applicable in certain fields such as $\mathbb{Z}_2$, because if there is a zero share, participants may infer that the secret value is zero.

For convenience, let x denote the secret share set $[x_1, x_2, \cdots, x_n]$ generated by ASS, and let $\langle u \rangle$ denote the secret share set $\langle u_1 \times u_2 \times \cdots \times u_n \rangle$ generated by MSS. Here, $[x_i]$ and $\langle u_i \rangle$ denote the shares held by participants S_i, respectively.

3.3. Pedersen Commitment

The Pedersen commitment scheme [26] provides a cryptographic method to guarantee both data integrity and verifiability. It enables the sender to convince the verifier of the existence of a hidden value. This scheme offers perfect hiding and computational binding properties [27], and its protocol consists of three main stages:

(1) *Initialization*: The trust center chooses a multiplicative group $\mathbb{G}$ of order q. The values g, h are the generating elements of $\mathbb{G}$.
(2) *Commitment*: Given a message $m \in \mathbb{Z}_q$, the committing party chooses a random number $r \in \mathbb{Z}_q$ as the blinding factor, then computes the commitment $com(m, r) = g^m h^r$ and sends it to the verifier.
(3) *Verification*: After the promiser discloses (m, r), the verifier computes $g^m h^r$ and determines whether it is equal to $com(m, r)$, so as to come to confirm whether the promiser really owns the message m or not.

The cryptographic hash function used in this protocol is SHA-256.

4. Problem Statement

4.1. System Architecture

Figure 1 illustrates that the TD-RCRF system framework embraces four key entities: a trust authority (TA), a data requester, two cloud servers, and multiple mobile users. Specifically, the TA handles system initialization tasks, such as allocating reputation values to mobile users upon joining the MCS system.

The cloud server receives task requests from data requesters and subsequently distributes them to mobile users. Upon completing their assignments, users upload encrypted shares of their perceived data. To ensure data reliability, the server verifies whether users' reputation values satisfy the task-specific credibility thresholds.

Proceeding further, the cloud servers compute the base truth shares based on the sensing data provided by users and continue iterating the process. Once the convergence condition is met, the truth shares are sent to the data requester, who uses it to assess the quality of the sensing data.

The trust authority updates the reputation values of the selected mobile users based on the data quality.

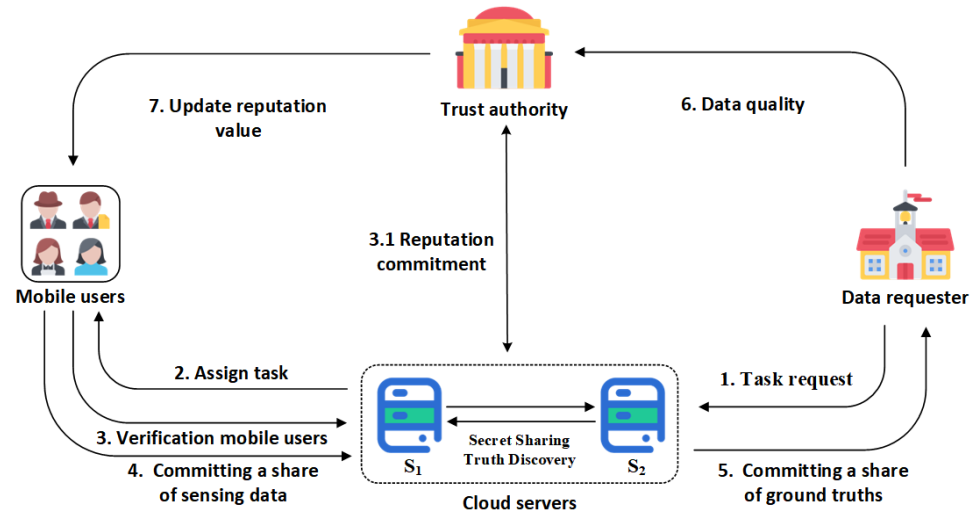


Figure 1. System model.

Specifically, the TD-RCRF framework includes the following steps:

Step 1 and 2. The data requester submits a task request, and the cloud server broadcasts the sensing task to mobile users.

Step 3. The cloud server verifies the mobile user's reputation value. If it has not been tampered with and falls within the specified range, the user is selected; otherwise, they are not selected.

Step 4. Mobile users participating in the sensing task send sensing data to the cloud server.

Step 5. The cloud server collaboratively calculates the ground truth share of the sensing task. After meeting the iteration requirements, the cloud server sends the final ground truth share to the data requester.

Step 6. The data requester evaluates the quality of the mobile users' sensing data and sends the results to a trust authority.

Step 7. The trust authority updates the reputation value based on the evaluation results, calculates the reputation commitment, and sends it to the user (see Figure 2).

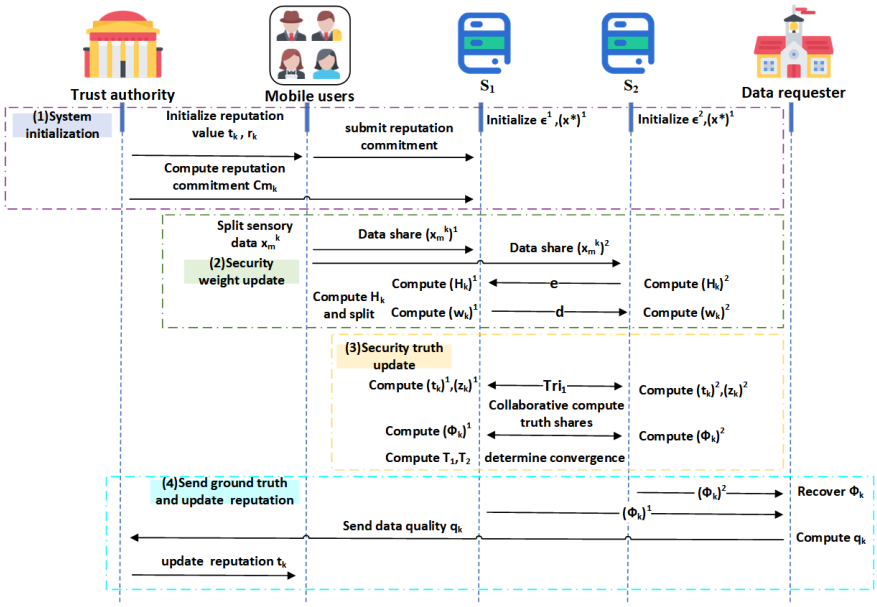


Figure 2. The workflow of TD-RCRF

4.2. Security Model

The goal of privacy protection is to safeguard privacy throughout the entire process while meeting accuracy requirements. In our system, it is essential to maintain a high level of privacy for users' perceptual data, reliability, and true estimate values. Additionally, the true values received by the query should fall within the small error margin specified by the accuracy requirements.

In practical applications, trust authorities are typically authoritative and trustworthy entities that do not collude with other entities, such as government agencies. Data requesters and cloud servers (S_1 and S_2), acting as honest yet curious entities, strictly adhere to the protocol without colluding with each other or malicious entities. However, they may attempt to uncover private information related to sensing tasks and mobile users. For instance, cloud servers might try to infer users' private data for additional benefits, deduce their identities from reputation values, or even disclose mobile users' weight values during the truth discovery process.

Mobile users, acting as data providers, are responsible for gathering perception data. This paper posits that the majority of mobile users are honest, though a minority may resort to malicious actions. These malicious users may submit false reputation values for economic gain to meet the reputation requirements of the perception task, thereby carrying out reputation tampering attacks. Additionally, they might deduce the reputation values and perception data of other trustworthy users, thus resulting in privacy inference attacks. However, we posit that mobile users will not deliberately tamper with their perception data to undermine the system. This is because countering such threats requires user self-authentication, and current tech like zero-knowledge proof and bilinear pairings can efficiently resolve this.

4.3. Design Goals

(1) Privacy Goals

This study aims to construct a privacy-preserving truth discovery framework that ensures both high accuracy and stable convergence. The proposed TD-RCRF approach is expected to follow the following privacy protection objectives:

Sensing Data Privacy. The mobile user's sensory data should not be disclosed to cloud servers and other staff, and the sensed data should not be accessible to any entity other than the mobile user themselves and the data requester.

Reputation Value Privacy. Reputation value privacy ensures that a user's reputation cannot be exposed, deduced, or associated with their identity by untrusted entities or potential attackers.

Weighting Privacy. Weighting privacy means that adversaries do not have access to the weighting values of each mobile user.

Ground Truth Privacy. Baseline truth value privacy means that the baseline truth value information for each iteration of sensory data should not be disclosed to staff and unrelated cloud servers.

(2) Security Goals

The TD-RCRF scheme needs to protect the perceived data, real identity, and reputation values of mobile users by preventing inference attacks and reputation-linking attacks initiated by the data requester and the cloud server. Moreover, the TD-RCRF scheme should stop malicious users from altering their reputation values and guarantee their authenticity.

(3) Accuracy And Practicality

The TD-RCRF scheme generates a baseline truth value for the sensing task that should be more accurate than other existing TD frameworks. In addition, to ensure the scheme's practical feasibility, mobile users' communication and computational overhead needs to be minimized.

5. Design of Our Scheme

In this section, we first list the formal symbols used in the scheme. Then, we introduce the basic algorithm for constructing the TD-RCRF scheme and its main stages in order.

5.1. Basic Algorithm Construction

5.1.1. Reputation Value Verification Algorithm

To facilitate description and subsequent content, Table 1 presents the formal symbols of the TD-RCRF scheme.

Table 1. Notations and descriptions.

Notation	Description
M	Total number of sensing missions
K	Number of mobile subscribers
u_k	Kth mobile subscriber
ω_k	Weight of the Kth user
x_m^*	Baseline truth value for task T_m
x_m^k	Sensing data for task T_m uploaded by user u_k
t_k	Reputation value of user u_k
Cm^k	Reputation commitment of user u_k
q_m^k	Perceived data quality for task x_m^k
T_m	mth sensing task
$[st(k)]_i$	Share of Difference in Perceived Value of User u_k
$[st^*]_i$	Sum of all user u_k variance shares

A zero-knowledge proof typically involves two participants: a prover and a verifier. The prover demonstrates to the verifier possession of knowledge regarding message n without disclosing additional details about it. This technique can be applied to verify the authenticity of a mobile user's reputation value to determine if it has been tampered with.

Specifically in the TD-RCRF scenario, cloud server S_1 serves as the verifier, contrasting with the mobile user as the prover. The latter must demonstrate to S_1 the authenticity of their supplied reputation value $t'_k \in \mathbb{Z}_p^*$, proving equivalence to $t_k \in \mathbb{Z}_p^*$, stored on S_1. Throughout this process, the prover does not reveal the reputation value t'_k. Based on the above definitions, we construct the basic structure of the RVVA algorithm based on the DLP and DDH problems.

We let l_1, l_2, s_1, s_2 be the four security parameters, and let $\mathbb{G}$ be a cyclic group of order p, where p is a large prime number and g_1 is an element in $\mathbb{Z}_p^*$. Further, let g_2, h_1, h_2 be an element in the generating group g_1, which will prevent the mobile user from getting the discrete logarithm with g_1 as the base, the discrete logarithm of g_1 with h_1 as the base, the discrete logarithm of h_2 with g_2 as the base, and the discrete logarithm of g_2 with h_2 as the base. We assume $t_k \in \mathbb{Z}_p^*$ as the reputation value assigned to the mobile user by the reputation center, and $t'_k \in \mathbb{Z}_p^*$ as its counterpart submitted by the same user during verification. The implementation steps are shown as follows:

(1) A user who wants to participate in the sensing task will choose random integers ω, η_1, η_2, from $\left[1, 2^{l_1+l_2}p+1\right]$, $\left[1, 2^{l_1+l_2+s_1}p-1\right]$, and $\left[1, 2^{l_1+l_2+s_2}p-1\right]$, respectively,

and then compute the commitments $Cm_1 = g_1^{\omega} h_1^{\eta_1} \bmod p$ and $Cm_2 = g_2^{\omega} h_2^{\eta_2} \bmod p$, and compute the hash function $H = Hash(Cm_1 \| Cm_2)$ of the commitments.

(2) These mobile users will then calculate the commitment $Cm'_k = g_2^{t'_k} h_2^{r_2} \bmod p$, $D = \omega + H \cdot t'_k$, $D_1 = \eta_1 + H \cdot r_1$, $D_2 = \eta_2 + H \cdot r_2 (r_1, r_2 \xleftarrow{R} [-2^{s_1} p + 1, 2^{s_1} p - 1])$, then send $\{H, D, D_1, D_2, Cm'_k\}$ to the cloud server S_1.

(3) When the server receives the $\{H, D, D_1, D_2, Cm'_k\}$ sent by the mobile user, S_1 will match the reputation commitment $Cm_k = g_1^{t_k} h_1^{r_1} \bmod p$ belonging to the mobile user from the reputation commitments issued by the reputation center during the system establishment phase. Per prior conditions, $t_k \in \mathbb{Z}_p^*$ constitutes the mobile user's reputation value, $r_1 \xleftarrow{R} [-2^{s_1} p + 1, 2^{s_1} p - 1]$.

(4) Finally, the cloud server can determine whether a mobile user's reputation commitment has been tampered with by using the following equation:

$$H_k = Hash(g_1^{D_k} h_1^{D_{k_1}} (Cm_k)^{-H_k} \bmod p \| g_2^{D_k} h_2^{D_{k_2}} (Cm'_k)^{-H_k} \bmod p) \quad (4)$$

If the equation holds, S_1 confirms that the mobile user's reputation value has not been tampered with and subsequently executes the privacy-preserving reputation value range proof algorithm. Conversely, the mobile user will not be selected, and the user's perceived data will be rejected. To simplify the subsequent discussion, this paper defines the reputation verification algorithm as RVVA, which is used to check whether the reputation value of a mobile user's u_k has been tampered with. We denote the reputation verification function by $\varphi(t, t') = \text{RVVA}(k, t, t')$, where t and t' represent points in time, respectively, and K denotes the Kth mobile user. If the user's reputation value is verified, the function RVVA(k,t, t') returns a value of 1. Conversely, if the reputation value is not verified, the return value is 0.

5.1.2. Multiplicative Secret Resharing Algorithm

We introduce two novel algorithms, MSRA and ASRA, for switching sharing secrets between multiplicative secret sharing (MSS) and additive secret sharing (ASS). In this section, we specify the multiplicative secret resharing algorithm. We need to execute the truth discovery algorithm on two cloud servers, each of which owns only a partial share of the user-aware data. The execution of truth discovery algorithms requires secure addition, multiplication, division, and logarithmic operations. While additive secret-sharing algorithms enable secure addition and multiplication operations, division and logarithmic operations are difficult to perform, relying solely on additive secret-sharing techniques.

Multiplicative secret resharing is a method of converting shared secrets on MSS to shared secrets on ASS, much like the idea of multiplicative secret sharing, which is essentially arithmetic $(g_1 + g_2) \leftarrow (x_1 + x_2) \cdot (y_1 + y_2)$, where the input to each server S_i is $\{x_i, y_i\}$. Multiplicative secret resharing is essentially the computation $(x_1 + x_2) \leftarrow (u_1 \cdot u_2)$, where the input to each server S_i is u_i. The outputs of these two parties are isomorphic, and in terms of the server's inputs, u_1 can be thought of as $(x_1 + x_2)$, and u_2 as $(y_1 + y_2)$.

In this scenario, server S_1, in this case, has x_1 and x_2 instead of x_1 and y_1, and the assignment of Beaver triples needs to be adjusted. Specifically, the generated Beaver triples are a, b, and c. The value c is assigned in the usual way in the offline phase through additive secret sharing. However, the values a and b will be assigned to S_1 and S_2, respectively, instead of being shared.

For S_1, the user holds x_1 and x_2 and wants to obtain a_1 and a_2, i.e., S_1 has u_1, a, and c_1, and S_2 has u_2, b, and c_2. As shown in Algorithm 1, S_1 will first compute $d \leftarrow u_1 - a$, which can be regarded as a $[[e]] \leftarrow [[y]] - [[b]]$ in the ternary, and S_2 will compute $e \leftarrow u_2 - b$. Then, the two servers interact with the values of d and e. S_1 computes $x_1 \leftarrow c_1 + e \cdot a$ and

S_2 computes $x_2 \leftarrow c_2 + d \cdot b + e \cdot d$, which can be regarded as the triad $d \cdot b + e \cdot d$. This step can be considered as $[[c]] + d \cdot [[b]] + e \cdot [[a]] + e \cdot d$ in the triad.

5.1.3. Additive Secret Resharing Algorithm

The additive secret resharing algorithm can be considered as an inverse process of multiplicative secret resharing, where the Beaver triples are distributed in the same way as in Algorithm 1. As shown in Algorithm 2, firstly, cloud servers S_1 and S_2 compute $[[t]] \leftarrow [[x]] - [[c]]$ and we let server S_2 compute so that e can be recovered by the equation $e \leftarrow t_1/a$. When S_2 receives e, sent by S_1, the share of u_2 can be recovered by calculating $u_2 \leftarrow e + b$, and the other value can be recovered by $d \leftarrow t_2/u_2$. Finally, S_1 receives d and computes $u_1 \leftarrow d + a$.

Algorithm 2 Multiplication Secret Resharing Protocol $[\![x]\!] \leftarrow MulSecRes(\langle u \rangle)$

Input: Shared secret $\langle u \rangle$ over MSS. Beaver triple: $a, b, [\![c]\!]$, where $c = a \cdot b$. The value a is shared by ASS, meanwhile a is held by S_1 and b is held by S_2.
Output: Shared secret $[\![x]\!] = (x_1 + x_2)$ over ASS such that $x_1 + x_2 = u$.
1: S_1 computes $d \leftarrow u_1 - a$.
2: S_2 computes $e \leftarrow u_2 - b$.
3: S_1 sends d to S_2.
4: S_2 sends e to S_1.
5: S_1 computes $x_1 \leftarrow c_1 + e \cdot a$.
6: S_2 computes $x_2 \leftarrow c_2 + d \cdot b + e \cdot d$.
7: **Return** $[\![x]\!]$.

5.2. Main Phases of TD-RCRF

System setup phase: Whenever a new user u_k enters the system, the reputation center computes a reputation commitment. During system initialization, a reputation commitment $Cm_k = g_1^{t_k} h_1^{r_k} \bmod p$ is computed by the reputation center for each newly registered user u_k, where the reputation value $t_k \in \mathbb{Z}_p^*$ and a random number $r_k \xleftarrow{R} [-2^{s_1}p + 1, 2^{s_1}p - 1]$, and then the reputation center sends $\{t_k, r_k\}$ to u_k over a secure channel to deliver the Cm_k securely to the cloud server S_1. The data requester initializes the truth discovery threshold ϵ, the reputation threshold t_0, the number of mobile users K required to limit the tasks, and the baseline truth value $\Phi^* = \{x_m^*\}_{m=1}^M$. The truth discovery threshold ϵ, the baseline truth value $\{x_m^*\}_{m=1}^M$, and the reputation threshold t_0 are divided into two shares, $([\epsilon]_1, [\epsilon]_2), ([\Phi^*]_1, [\Phi^*]_2), ([t_0]_1, [t_0]_2)$, respectively, using the additive secret sharing scheme $SS(\cdot)$. The shares $\{[\epsilon]_i, [\Phi^*]_i, [t_0]_i\}$ are then submitted to the cloud server S_i, respectively (we assume that all tasks have a reputation value of t_0).

The initial baseline truth $\{x_m^*\}_{m=1}^M$ is generated and provided by the data requester during system initialization. The data requester randomly determines the initial value, which is then split into two additive shares $[x_m^*]_1$ and $[x_m^*]_2$ and sent to cloud servers S_1 and S_2, respectively. The truth discovery algorithm starts from this initial value and iterates until convergence.

(1) Task request submission and sensing task allocation
The data requester generates M sensing tasks $\{T_m\}_{m=1}^M$ with a unique identifier $\{ID_m\}_{m=1}^M$ for each task and divides the tasks into shares $[\{T_m\}_{m=1}^M]_1, [\{T_m\}_{m=1}^M]_2$ via the secret sharing scheme $SS(\cdot)$. Then, the sets $\{[\{T_m\}_{m=1}^M]_1, \{ID_m\}_{m=1}^M\}$ and $\{[\{T_m\}_{m=1}^M]_2, \{ID_m\}_{m=1}^M\}$ are sent over a secure channel to cloud servers S_1 and S_2, respectively, and $\{T_m\}_{m=1}^M$ is sent to the reputation center TA. Finally, the server broadcasts the share of the perceived task to the mobile user u_k that wants to participate in the task.

(2) Verify and select mobile users
When these users receive the sensing task shares $\{[\{T_m\}_{m=1}^M]_i, \{ID_m\}_{m=1}^M\}$, they can sum the corresponding shares to obtain $\{T_m\}_{m=1}^M$ according to the nature of addi-

tive secret sharing. Firstly, mobile users need to compute $Cm_{k_1} = g_1^{\omega_k} h_1^{\eta_{k_1}} \bmod p$, $Cm_{k_2} = g_2^{\omega_k} h_2^{\eta_k} \bmod p$, and $H_k = Hash(Cm_{k_1} \| Cm_{k_2})$, where $\omega_k \xleftarrow{R} [1, 2^{l_1+l_2} p + 1]$, $\eta_{k_1} \xleftarrow{R} [1, 2^{l_1+l_2+s_1} p - 1]$, $\eta_{k_2} \xleftarrow{R} [1, 2^{l_1+l_2+s_2} p - 1]$. Next, mobile user u_k computes $Cm'_k = g_2^{t'_k} h_2^{r'k} \bmod p, D_k = \omega_k + H_k \cdot t'_k, D_{k_1} = \eta_{k_1} + H_k \cdot r_k, D_{k_2} = \eta_{k_2} + H_k \cdot r'_k$, where $r'_k \xleftarrow{R} [-2^{s_1} p + 1, 2^{s_1} p - 1]$. Finally, u_k will submit $\{u_k, \{T_m\}_{m=1}^M, H_k, D_k, D_{k_1}, D_{k_2}, Cm'_k\}$ to the server S_1 . Then, the cloud server S_1 can verify the commitment Cm'_k via Equation (4), where t'_k is the reputation value submitted by the mobile user and $t'_k = t_k$ if the reputation value has not been tampered with.

For K users, when the reputation verification algorithm $\mathrm{RVVA}(k, t, t') = 1$, it indicates that all users have passed the reputation verification. In this case, the cloud server S_1 will send a set containing the identifiers $\{u_k\}_{k=1}^K$ of all verified mobile users to the reputation center (TA). Next, the reputation center will send the corresponding user's reputation value t_k to the server S_i in the form of a share. Upon receiving the reputation value share, the server can invoke the reputation value range comparison algorithm designed by MSRA and ASRA to verify that the user's reputation value meets the range requirements of the sensing task. Specifically, Algorithm 3 shows in detail the process of reputation value tampering detection and range verification.

Algorithm 3 Reputation tampering detection and range proofing

Input: $\{u_k\}_{k=1}^K, \{T_m\}_{m=1}^M$, Shared secret $[\![t_k]\!]$, $[\![t_0]\!]$, and $[\![t_{\max}]\!]$.
Output: 1 if $t_0 \le t_k \le t_{\max}$, 0 if $t_0 \not\le t_k \not\le t_{\max}$.

Mobile user
for $k = 1, 2, \cdots, K$ **do**
  Select random integers $\omega_k, \eta_{k_1}, \eta_{k_2}$ and r'_k
  Compute $Cm'_k = g_2^{t'_k} h_2^{r'_k} \bmod p, Cm_{k_1} = g_1^{\omega_k} h_1^{\eta_{k_1}} \bmod p$, $Cm_{k_2} = g_2^{\omega_k} h_2^{\eta_{k_2}} \bmod p$ and $H_k = Hash(Cm_{k_1} \| Cm_{k_2})$
  Compute $D_k = \omega_k + H_k \cdot t'_k, D_{k_1} = \eta_{k_1} + H_k \cdot r_k$ and $D_{k_2} = \eta_{k_2} + H_k \cdot r'_k$
Send $\{u_k, \{T_m\}_{m=1}^M, H_k, D_k, D_{k_1}, D_{k_2}, Cm'\}_{k=1}^K$ to S_1
for $k = 1, 2, \cdots, K$ **do**
  Compute $H_k = Hash(g_1^{D_k} h_1^{D_{k_1}} (Cm)^{-H_k} \| g_2^{D_k} h_2^{D_{k_2}} (Cm'_k)^{-H_k}) \bmod p$
for $k = 1, 2, \cdots, K$ **do**
  if $RVVA(k, t, t') = 1$ **then** S_1 sends $\{u_k\}_{k=1}^K$ to the TA
TA: Obtains reputation value of K mobile users from the database
Splitting the user's reputation value into two shares through $SS(\cdot)$
TA sends $\{[t_k]_i, u_k\}$ to S_i
S_i: S_1 computes $d_1 = [t_k]_1 - [t_0]_1, g_1 = [t]_1 - [t_{\max}]_1$
S_2 computes $d_2 = [t_k]_2 - [t_0]_2, g_2 = [t]_2 - [t_{\max}]_2$
S_1 and S_2 collaboratively compute $\langle u \rangle \leftarrow AddSecRes([\![d]\!])$, $\langle v \rangle \leftarrow AddSecRes([\![g]\!])$
S_1 and S_2 reveal the signs of u_1, u_2 and v_1, v_2
if ($u_1 < 0$ and $u_2 > 0$) or ($u_1 > 0$ and $u_2 < 0$) **then Return** 0.
else if ($v_1 < 0$ and $v_2 > 0$) or ($v_1 > 0$ and $v_2 < 0$) **then Return** 0.
else **Return** 1.
end if

(3) Submission of sensing data

When the user u_k passes the reputation value tampering and range verification, the user senses the data of the task and uploads the data. Each user u_k also splits the sensed value $\{x_m^k\}_{m=1}^M$ into two copies $(\{[x_m^k]_{m=1}^M\}_1, \{[x_m^k]_{m=1}^M\}_2)$ via the additive secret sharing scheme $SS(\cdot)$. Each user u_k, in addition to encrypting the sensed data, randomly generates two multiplicative triples $Tri_1 = (a_1, b_1, c_1)$ and $Tri_2 = (a_2, b_2, c_2)$, divides them into two-part shares using $SS(\cdot)$, and divides c_2 into two-part shares. Finally, the user u_k generates a random unique identifier ID_k, submits the share $\{\{[x_m^k]_{m=1}^M\}_i, ID_k\}, [Tri_1]_i, [c_2]_i\}$ to the cloud server S_i via a secure channel, and sends a_1 to the cloud server S_1 and b_2 to the cloud server S_2, respectively.

(4) Perform truth discovery and send baseline truth values

(1) Security weight update on the cloud server side

Based on the initialized truth shares $[\Phi^*]_i$, each cloud server S_i can now update the weights of each user u_k accordingly. The cloud server S_i first calculates the difference share of the user-perceived values $[st(k)]_i \leftarrow \sum_{m=1}^{M} \left[x_m^k\right]_i - [x_m^*]_i$, and then calculates the sum of the difference shares of all user-perceived values $[st^*]_i \leftarrow \sum_{k=1}^{K} [st(k)]_i$, after which S_i invokes the secure division protocol $SecDiv([[st^*]], [[st(k)]], 1, -1)$ shown in Algorithm 4 and secure logarithmic protocol $SecLog([[H_k]], e)$ shown in Algorithm 5.

Algorithm 4 Secure division protocol $[\![H_k]\!] \leftarrow SecDiv([\![st^*]\!], [\![st(k)]\!], 1, -1)$

Input: Shared secrets $[\![st^*]\!]$, $[\![st(k)]\!]$ and public integers 1,-1.
Output: Shared secret $[\![H_k]\!]$ such that $H_k = [\![st^*]\!]^1 \cdot [\![st(k)]\!]^{-1}$.
1: S_1 and S_2 collaboratively compute $\langle u \rangle^1 \leftarrow AddSecRes([\![st^*]\!])$ and $\langle u \rangle^2 \leftarrow AddSecRes([\![st(k)]\!])$.
2: S_1 computes $v_1 \leftarrow \left(u_1^1\right)^1 \cdot \left(u_1^2\right)^{-1}$.
3: S_2 computes $v_2 \leftarrow \left(u_2^1\right)^1 \cdot \left(u_2^2\right)^{-1}$.
4: S_1 and S_2 collaboratively compute $[\![H_k]\!] \leftarrow MulSecRes(\langle v \rangle)$.
5: **Return** $[\![H_k]\!]$.

Algorithm 5 Secure logarithm protocol $[\![w_k]\!] \leftarrow SecLog([\![H_k]\!], e)$

Input: Shared secret $[\![H_k]\!]$ and public base number e.
Output: Shared secret $[\![w_k]\!]$ such that $w_k = \log_e^{|H_k|}$.
1: S_1 and S_2 collaboratively compute $\langle u \rangle \leftarrow AddSecRes([\![H_k]\!])$.
2: S_1 computes $[w_k]_1 \leftarrow log_e^{|u_1|}$.
3: S_2 computes $[w_k]_2 \leftarrow log_e^{|u_2|}$.
4: **Return** $[\![w_k]\!]$.

The cloud server S_i finally gets the weight share $[[w_k]]$ with weight $w_k = [w_k]_1 + [w_k]_2$.

(2) Secure truth updates on the cloud server side

In the truth update phase, each cloud server S_i only has the weighted value share, i.e., $[w_k]_i$, and the perceived value share, i.e., $\{[x_m^k]_{m=1}^M\}_i, ID_k$, for each user's u_k. We need to invoke the triad M to compute the weighted value share of each user u_k: $[w_k \cdot x_m^k]_i$. Based on the share of perceived values of all users $\{[x_m^k]_{m=1}^M\}_{k=1}^K$, S_i can compute the share of the weighted sum of the perceived values $[t]_i$ as follows:

$$[t]_i \leftarrow \sum_{k=1}^{K} Mul_{tri_1}([x_m^k]_i, [w_k]_i) \tag{5}$$

The two cloud servers will use the values submitted by each user u_k as the operands in this multiplication operation. The cloud server S_i will denote the share of the weight sum as $[z]_i \leftarrow \sum_{k=1}^{K} [w_k]_i$ and use the secure addition resharing protocol AddSecRes ($[[x]]$), the secure multiplication resharing protocol $MulSecRes(\langle u \rangle)$, and the secure division protocol $SecDiv([[t]], [[z]], 1, -1)$ to compute the baseline truth value Φ, as shown in Algorithms 6 and 7:

Algorithm 6 Secure division protocol $[\![\Phi]\!]^{I_m} \leftarrow SecDiv([\![t]\!],[\![z]\!],1,-1)$

Input: Shared secrets $[\![t]\!]$, $[\![z]\!]$ and public integers 1,-1.
Output: Shared secret $[\![\Phi]\!]$ such that $\Phi = [\![t]\!]^1 \cdot [\![z]\!]^{-1}$.
1: S_1 and S_2 collaboratively compute $\langle u\rangle^1 \leftarrow AddSecRes([\![t]\!])$ and $\langle u\rangle^2 \leftarrow AddSecRes([\![z]\!])$.
2: S_1 computes $v_1 \leftarrow \left(u_1^1\right)^1 \cdot \left(u_1^2\right)^{-1}$.
3: S_2 computes $v_2 \leftarrow \left(u_2^1\right)^1 \cdot \left(u_2^2\right)^{-1}$.
4: S_1 and S_2 collaboratively compute $[\![\Phi]\!]^{I_m} \leftarrow MulSecRes(\langle v\rangle)$.
5: **Return** $[\![\Phi]\!]$.

Cloud server S_i finally gets the benchmark truth share $[[\Phi]]$, first cloud servers S_1 and S_2 compute $T_1 \leftarrow \Phi_1^{I_m} - \Phi_1^{I_{m-1}}$, $T_2 \leftarrow \Phi_2^{I_m} - \Phi_2^{I_{m-1}}$, respectively, and then the two servers run the comparison protocol SecCom ($[[T]]$,$[[\epsilon]]$) collaboratively. If the output is 1, then it is decided that $T < \epsilon$, and the servers will send the shares Φ_1 and Φ_2 as well as $\left[x_m^k\right]_1$ and $\left[x_m^k\right]_2 (1 \le k \le K, 1 \le m \le M)$ to the data requesters, respectively. If the output is 0, then it is decided that $T > \epsilon$ or $T = \epsilon$. At this point, the servers will continue the iterative computation until the truth value converges. The truth shares can recover the baseline truth value $\Phi = \Phi_1 + \Phi_2 = \frac{t}{z}$.

$$\Phi = \frac{([t_1] + [t_2])}{([z_1] + [z_2])} = \frac{\sum_{k=1}^{K}([w_k \cdot x_m^k]_1 + [w_k \cdot x_m^k]_2)}{\sum_{k=1}^{K}([w_k]_1 + [w_k]_2)} \tag{6}$$

Algorithm 7 Secure comparison protocol $SecCom([\![T]\!],[\![\epsilon]\!])$

Input: Shared secrets $[\![T]\!]$ and $[\![\epsilon]\!]$.
Output: If $T < \epsilon$, return 1; if $T > \epsilon$ or $T == \epsilon$, return 0.
1: S_1 computes $d_1 \leftarrow T_1 - \epsilon_1$.
2: S_2 computes $d_2 \leftarrow T_2 - \epsilon_2$.
3: S_1 and S_2 collaboratively compute $\langle u\rangle \leftarrow SecAddRes([\![d]\!])$.
4: S_1 and S_2 reveal the signs of u_1 and u_2.
5: **if** ($u_1 < 0$ and $u_2 > 0$) or ($u_1 > 0$ and $u_2 < 0$) **then**
6: **Return** 1.
7: **else**
8: **Return** 0.
9: **end if**

(5) Assessment of data quality

The data requester receives $\{\Phi_1, \Phi_2, [x_m^k]_1, [x_m^k]_2\}$, obtains the final baseline truth value $\{x_m^{I_m}\}_{m=1}^{M}$, and evaluates the submitted sensed data $\{x_m^k\}(where 1 \le k \le K,\ 1 \le m \le M)$. The requester calculates the difference between x_m^k and $x_m^{I_m}$: $d_m^k = d(x_m^k, x_m^{I_m})$. Here, d_m^k is used to measure the similarity between the two data x_m^k and $x_m^{I_m}$, where a smaller $d(x_m^k, x_m^{I_m})$ means that the two data are more similar, and then calculate the quality of the sensed data x_m^k [28], as follows:

$$q_m^k = \frac{\frac{1}{d_m^k + \xi}}{\sum_{k=1}^{K} \frac{1}{d_m^k + \xi}} \tag{7}$$

Here, ξ serves as a tunable constant governing sensed data quality deviation, with the full evaluation logic formalized in Algorithm 8.

Algorithm 8 Quality evaluation

Require: Input: $\left\{x_m^k\right\}_{k=1}^K$, ξ, and $x_m^{I_m}$.

Ensure: Output: Data quality set $\left\{q_m^k\right\}_{k=1}^K$.

1: **for** $k = 1, 2, \cdots, K$ **do**
2: **for** $m = 1, 2, \cdots, M$ **do**
3: $d_m^k = d\left(x_m^k, x_m^{I_m}\right)$;
4: **end for**
5: $q_m^k = \frac{1}{d_m^k + \xi} / \sum_{j=1}^K \frac{1}{d_m^j + \xi}$;
6: **end for**
7: **Return** $\left\{q_m^k\right\}_{k=1}^K$.

Finally, the data requester sends $\{q_m^k\}_{k=1}^K$ to the TA.

(6) Renewal of reputation value After the TA receives $\{q_m^k\}_{k=1}^K$, it updates the reputation value of the mobile user according to the reputation update algorithm, which has the process of updating [29]

$$t_k = A + \frac{(B - A)}{\left(1 + D \cdot e^{-F \cdot (q_m^k - M)}\right)^{\frac{1}{\tilde{h}}}} \tag{8}$$

The reputation update algorithm takes $A = 0, B = 1, D = 1, F = 1, M = 1, \tilde{h} = 1$. Then, $t_k = \frac{1}{(1 + \cdot e^{-(q_m^k - 1)})}$. The detailed process of reputation update is shown in Algorithm 9. Finally, the TA sends the updated reputation value to the user u_k.

Algorithm 9 Reputation update

Require: Input: $\left\{q_m^k\right\}_{k=1}^K$.

Ensure: Output: Reputation values $\{t_k\}_{k=1}^K$.

1: Set initial reputation value as t_k;
2: **for** $k = 1, 2, \ldots, K$ **do**
3: **for** $m = 1, 2, \ldots, M$ **do**
4: $t_k = \frac{1}{\left(1 + e^{-\left(q_m^k - 1\right)}\right)}$; $t_k = t_k \cdot q_m^k$;
5: **end for**
6: **end for**
7: **Return** $\{t_k\}_{k=1}^K$.

6. Security Definition and Proof

This section proves the security and feasibility of the algorithms and protocols in the TD-RCRF scheme.

6.1. Privacy Analysis

6.1.1. Reputational Value Privacy

Theorem 1. *As the reputation value is encapsulated through a commitment exchanged between user u_k and the trusted authority (TA), the TD-RCRF framework ensures that this value remains confidential. That is, no probabilistic polynomial-time (PPT) adversary A is able to infer or extract user u_k's reputation.*

Proof. If any PPT adversary A attempts to reveal the reputation value t_k of the mobile user u_k through the reputation commitments Cm_k and Cm'_k, A must make these commitments public. Further, we need to consider two different scenarios:

1. Once attacker A acquires the reputation commitment Cm_k, any $t_k \in \mathbb{Z}_p^*$ and a randomly and uniformly chosen $r_k \in \mathbb{Z}_p^*$, Cm_k is uniformly distributed over the finite field $\mathbb{G}_p$. When $t_k \neq t_{k_1}\left(t_k, t_{k_1} \in \mathbb{Z}_p^*\right)$, t_{k_1} is chosen by adversary A to uncommit Cm_k, and $Cm_k = Cm_{k_1}, r_k \neq r_{k_1}$, we can compute

$$\frac{Cm_k}{Cm_{k_1}} = g_1^{t_k - t_{k_1}} \cdot h_1^{r_k - r_{k_1}} \bmod p = 1 \rightarrow \log_{g_1}^{h_1} = \frac{t_k - t_{k_1}}{r_k - r_{k_1}} \bmod p \tag{9}$$

Assuming that the attacker A has access to the reputation value t_k, then theoretically, A will be able to compute $\log_{g_1}^{h_1}$. However, according to the analysis, in the framework of Pedersen's commitment, A can neither know nor solve for $\log_{g_1}^{h_1}$. Based on this assertion, A is unable to derive t_k from the reputation commitment when t_k is not equal to t'_k. Similarly, if A has access to another reputation commitment Cm'_k, we can also conclude that A is similarly unable to expose the reputation value t'_k from Cm'_k.

2. Suppose that the attacker A has obtained the reputation commitments Cm_k and Cm'_k and satisfies the condition $t_k = t_{k_1}$, but $r_k \neq r_{k_1}$. In this case, A can compute $\frac{Cm_k}{Cm'_k} = \frac{g_1^{t_k} \cdot h_1^{r_k}}{g_2^{t'_k} \cdot h_2^{r'_k}} \bmod p$. Considering that g_1 is an element of the modulo p multiplicative group $\mathbb{Z}_p^*$ and that g_2, h_1, h_2 are both elements of the g_1 generating group, we can assume that $g_2 = g_1^{\mu}, h_2 = h_1^{\mu'}$, where μ and μ' are both elements in $\mathbb{Z}_p^*$, can then be computed as follows:

$$\frac{Cm_k}{Cm'_k} = \frac{g_1^{t_k} \cdot h_1^{r_k}}{g_2^{t'_k} \cdot h_2^{r'_k}} \bmod p = g_1^{t_k - \mu \cdot t'_k} \cdot h_1^{r_k - \mu' \cdot r'_k} \bmod p \tag{10}$$

Define $t = t_k - \mu \cdot t'_k$, and $r = r_k - \mu' \cdot r'_k$ such that we have $\frac{Cm_k}{Cm'_k} = g_1^t \cdot h_1^r \bmod p$. Since r does not contain any information related to the reputation value t_k, even if the attacker A knows r, they will not be able to obtain any information about t_k from it. Furthermore, when A has r and wishes to derive t, they will need to solve the logarithmic problem $\log_{g_1}^{(\frac{Cm_k}{Cm'_k})}$. However, based on the literature, attacker A can neither determine nor solve the logarithmic problem in Pedersen's commitment, i.e., $\log_{g_1}^{(\frac{Cm_k}{Cm'_k})}$. As a result, A is not able to resolve the reputation values t_k and t_{k_1} from the sum of the reputation commitments Cm_k and Cm'_k. Similarly, A is not able to expose the reputation threshold value t_0. □

6.1.2. Sensing Data Privacy and Weighting Privacy

In this scheme, the sensing data and weights of the mobile user u_k are privacy-preserving.

Proof. For S_1 and S_2, only a uniform random share of the sensory data can be obtained. Assuming that the participants are not colluding, the cloud server cannot restore the plaintext data. The calculation of weights depends on the sensory data shares, and each server only holds a portion of the user's sensory data and cannot perform the calculation independently. In addition, the security and simulability of this scheme have been proven. Based on Lemma 1 and Definition 1, this scheme is secure. □

6.1.3. Secret Sharing of Privacy

In this section, we prove the security of the multiplicative secret resharing algorithm (MSRA) and the additive secret resharing algorithm (ASRA) in this scheme. Suppose

the attacker is allowed to corrupt only one cloud server (S_1 or S_2). Then, security can be guaranteed by simply proving that for given inputs and outputs, the corrupter's view is simulatable [30]. Specifically, this scheme uses the following definitions and theorems.

Definition 1. *Suppose that for any real-world adversary A, there exists a probabilistic polynomial-time simulator S, S that is able to simulate an ideal world view that is indistinguishable from the real-world view; then, the protocol π is considered to be UC-safe, or, in other words, it implements an ideal functional function in the UC sense of the term F. The key point in proving the UC safety of the protocol is to show that for any party (including the adversary), the incoming views consisting of messages and outputs sent by any other party are indistinguishable.*

Lemma 1. *Let $x \in \mathbb{F}$, and consider a uniformly random element $r \in \mathbb{F}$ that is independent of x. Under this condition, the value $x \pm r$ preserves the uniform distribution over $\mathbb{F}$ and remains statistically independent from x [31].*

Lemma 2. *In the case where the nonzero element r is uniformly distributed and independent of x, for any nonzero element $x \in \mathbb{F}$, the nonzero element $r \cdot x$ is uniformly distributed and independent of x [32].*

Proof. Suppose there exists a nonzero element $r \in \mathbb{F}$ that has the property of being uniformly distributed and independent of all the elements x in the set. Then, the product $x \cdot r$ also has the property of being uniformly distributed. This is because the mapping function $f_r(x) := x \cdot r$ is a bilinear mapping of the group $\mathbb{F}$. □

To further enhance the security analysis of the proposed scheme, we provide explicit simulation-based proof sketches for its core components. The simulator construction formally demonstrates that each component does not disclose any private information beyond the final output, thereby ensuring the privacy and security of the entire protocol.

Simulator for MSRA: The simulator S_{MSRA} generates random shares that are computationally indistinguishable from real shares. Since all intermediate values are uniformly distributed and independent, no information about private inputs is leaked.

Simulator for ASRA: The simulator S_{ASRA} produces valid additive and multiplicative shares without accessing any private data. Due to the security of secret sharing, the view of the simulator is identical to that of the real protocol.

Simulator for Weight Update: The simulator S_{Weight} computes weighted values using only public information. All internal values are randomized, so privacy is preserved.

Simulator for Truth Update: The simulator S_{Truth} simulates aggregation operations under secret sharing. No private information is disclosed during the whole procedure.

6.1.4. Security of the Multiplicative Secret Resharing Algorithm

Algorithm 2 exhibits correctness in a model based on the honest-but-curious principle and ensures that it is UC-safe.

Proof. To verify its safety, given that the participants S_1 or S_2 in Algorithm 2 are equivalently symmetric, it suffices to argue that the view of S_1 is simulatable. S_1 receives an input (e) and produces an output x_1, where $e = u_2 - b$, and $x_1 = c_1 + e \cdot a$. Given that the values b and c_1 are both uniformly distributed and independent of any private input, it is perfectly

feasible to construct a perfect simulator for S_1 according to Lemma 1. Correctness is given by the following equation:

$$\begin{aligned} x = x_1 + x_2 &= c + d \cdot b + e \cdot a + e \cdot d \\ &= c + u_1 \cdot \ b - a \cdot b + u_2 \cdot a - a \cdot b \\ &+ u_1 \cdot u_2 - u_2 \cdot a - u_1 \cdot \ b + a \cdot b \\ &= u_1 \cdot u_2 = u \end{aligned} \tag{11}$$

□

6.1.5. Security of the Additive Secret Resharing Algorithm

Algorithm 10 is correct and UC-safe in the honest-but-curious model.

Algorithm 10 Additive secret resharing algorithm $\langle u \rangle \leftarrow AddSecRes([\![x]\!])$

Input: Additive secret $[\![x]\!]$ over ASS. Beaver triple: $a, b, [\![c]\!]$, where $c = a \cdot b$. The value a is shared by ASS, meanwhile a is held by S_1 and b is held by S_2.
Output: Shared secret $\langle u \rangle = (u_1, u_2)$ over MSS such that $u_1 \cdot u_2 = x$.
1: S_1 computes $\alpha \leftarrow (x_1 - c_1)$.
2: S_1 sends α to S_2.
3: S_2 computes $\beta \leftarrow (x_2 - c_2)$.
4: S_2 sends $u_2 \leftarrow b + \beta$.
5: S_1 computes $u_1 \leftarrow a + \alpha$.
6: **Return** $\langle u \rangle$.

Proof. To prove the security of ASRA, we must show views of participants S_1 and S_2 can be modeled. For S_2, its incoming view is (e) and $e = (x_1 - c_1)/a$. For S_1, its incoming view is (d), where $d \leftarrow (x_2 - c_2)/u_2$. Since parameters $a, b, c_1\ c_2$ follow a uniform distribution and are independent of private inputs, according to Lemmas 1 and 2, the incoming views of both S_1 and S_2 can be simulated. Similarly, their output views can also be simulated. The correctness is verified by the following equation:

$$\begin{aligned} u = u_1 \cdot u_2 &= (d + a) \cdot (e + b) \\ &= x_2 - c_2 + a \cdot e + a \cdot b \\ &= x_2 - c_2 + x_1 - c_1 + a \cdot b \\ &= x_1 + x_2 = x \end{aligned} \tag{12}$$

□

6.2. Security Analysis

Given that the reputation authority TA and the mobile user u_k establish interactions with the cloud server through reputation commitment, attacker A can neither deduce the reputation value of u_k in the TD-RCRF nor tamper with it. This shows that the TD-RCRF is resilient to reputation inference and tampering attacks.

Proof. The reputation value t_k of user u_k is embedded in the reputation commitments $Cm_k = g_1^{t_k} h_1^{r_k} \bmod p$ and $Cm'_k = g_2^{t_k} h_2^{r_k} \bmod p$, $Cm'_k = g_2^{t'_k} h_2^{r'_k} \bmod p$, and although adversary A is able to capture the commitments Cm_k and Cm'_k by listening to the wireless channel, they are unable to extract any specific information about the reputation value t_k from it according to Theorem 1. Therefore, TD-RCRF shows its effectiveness in defending against reputation inference attacks. In addition, if a malicious user manipulates the reputation value when calculating the reputation commitment Cm'_k, the cloud server S_1 will reject the

sensing data submitted by them, as these data cannot pass the verification mechanism. This shows that TD-RCRF is effective in defending against reputation tampering attacks. □

6.3. Nature Analysis

In this section, we provide an exhaustive analysis of the accuracy, completeness, and zero-knowledge proof of the RVVA algorithm.

A mobile user u_k can pass the reputation verification process only if the two reputation commitments Cm_k and Cm'_k contain the same reputation value (i.e., t_k is equal to t'_k), which verifies the correctness of the RVVA algorithm.

Proof. In this algorithm, the user u_k will calculate $\{H_k, D_k, D_{k_1}, D_{k_2}, Cm'_k\}$ and later send it to the cloud server S_1, where $H_k = Hash(Cm_{k_1} \| Cm_{k_2}), D_k = \omega_k + H_k \cdot t'_k$, $D_{k_1} = \eta_{k_1} + H_k \cdot r_k$, $D_{k_2} = \eta_{k_2} + H_k \cdot r'_k, Cm'_k = g_2^{t'_k} h_2^{r'_k} \bmod p$, $r'_k \xleftarrow{R} [-2^{s_1}p + 1, 2^{s_1}p - 1]$, $\omega_k \xleftarrow{R} [1, 2^{l_1+l_2}p + 1], \eta_{k_1} \xleftarrow{R} [1, 2^{l_1+l_2+s_1}p - 1], \eta_{k_2} \xleftarrow{R} [1, 2^{l_1+l_2+s_2}p - 1]$.

The cloud server S_1 can receive the $Cm_k = g_1^{t_k} h_1^{r_k} \bmod p$ sent by the reputation center and then calculate and verify whether it is valid for both sides of Equation (4). The specific verification process is shown in Equation (13):

According to Equation (13), we learn that $H_k = Hash(Cm_{k_1} \| Cm_{k_2})$ only if t_k is equal to t'_k, thus allowing user u_k to successfully pass the reputation verification. This proves the accuracy of the RVVA algorithm.

$$\begin{aligned} & H(g_1^{D_k} h_1^{D_{k_1}} \cdot (Cm_k)^{-H_k} \bmod p \| g_2^{D_k} h_2^{D_{k_2}} \cdot (Cm'_k)^{-H_k} \bmod p) \\ & = H(g_1^{\omega_k + H_k \cdot t'_k} h_1^{\eta_{k_1} + H_k \cdot r_k} \cdot (g_1^{t_k} h_1^{r_k})^{-H_k} \bmod p. \\ & \| g_2^{\omega_k + H_k \cdot t'_k} h_2^{\eta_{k_2} + H_k \cdot r'_k} \cdot (g_2^{t'_k} h_2^{r'_k})^{-H_k} \bmod p) \\ & = H(g_1^{\omega_k + H_k \cdot (t'_k - t_k)} h_1^{\eta_{k_1}} \bmod p \| g_2^{\omega_k} h_2^{\eta_{k_2}} \bmod p) \\ & = H(g_1^{\omega_k + H_k \cdot (t'_k - t_k)} h_1^{\eta_{k_1}} \bmod p \| Cm_{k_2}) \end{aligned} \tag{13}$$

□

In the reputation verification phase, the attacker A cannot learn anything about the reputation values t_k and t'_k from the two reputation commitments Cm_k and Cm'_k. This point proves that the RVVA algorithm has the zero-knowledge property.

Proof. Suppose that A is a potential attacker, g_1 is an element of large prime order in $\mathbb{Z}_p^*$, and g_2, h_1, h_2 is an element in the generating group of g_1. A is able to be informed of the commitments Cm_k and Cm'_k of the reputation value t_k; however, according to Theorem 1, A is unable to derive any information about t_k and t'_k from these two commitments. This shows that the RVVA algorithm achieves zero-knowledge verification. □

7. Performance Evaluation

In this section, we first introduce the experimental setup and parameter configuration. Then, we analyze and evaluate the performance of the algorithm in TD-RCRF, including computational and communication overhead. Finally, we assess the truth discovery and reputation update mechanisms, benchmarking them against state-of-the-art approaches.

All hash operations in the proposed scheme are implemented using the SHA-256 cryptographic hash function, which provides strong collision resistance for security. We empirically validate TD-RCRF's efficacy through Python 3.8 experiments, benchmarking performance on an Intel(R) Core(TM) i5-8300H CPU @2.30GHz with 16.00 GB RAM. Experiments are run on 64-bit Windows 10. Secret sharing secures data exchanges among entities (e.g., cloud servers) and requesters. Execution times appear in Table 2.

Table 2. Calculation cost of basic cryptographic operations.

Symbol	Meaning	Times (ms)
T_{hg}	Time of a hash generation operation	2
T_{hv}	Time of a hash verification operation	0.6
T_e	Time of an exponential operation	10
T_{add}	Time of the ASRA algorithm	2.04
T_{mul}	Time of the MSRA algorithm	2.12
T_m	Time of a multiplication operation	0.015
T_{tri}	Time of multiplication triplets	1.8

In TD-RCRF, the disclosure parameter p and the output of the hash function $Hash(\cdot)$ are both 64 bits in length, denoted by $|p| = |H| = 64$. Meanwhile, the sequence index m, ground truth x_m^*, sensed value x_m^k, and user identity are all fixed-size elements represented using 32-bit values, i.e., $|c| = 32$.

7.1. Performance of the RVVA Algorithm

In this section, we will analyze the computational cost and communication overhead of RVVA one by one.

7.1.1. Computation Overhead

To prevent reputation forgery by dishonest mobile users, TD-RCRF adopts the RVVA protocol, which ensures data integrity and filters valid contributors. Within this framework, when K users are involved, S_1 performs K hash checks, incurring a computational load of $K \cdot T_{hv}$ and a total runtime of 0.6 Kms. Meanwhile, each user completes a sequence of four exponentiations, two multiplications, and one hash operation, amounting to a cost of $4T_e + 2T_m + T_{hg}$ and a runtime of 42.03 ms. Accordingly, the time complexity of S_1 under RVVA is O(K).

7.1.2. Communication Overhead

Under the RVVA protocol, each user u_k transmits $\{H_k, D_k, D_{k_1}, D_{k_2}, Cm'_k\}$ to server S_1, which checks the integrity of the user's reputation value. This verification is performed for all K users. Once all users successfully pass the check, S_1 forwards the set $\{u_k\}_{k=1}^K$ to the TA. Based on this process, the message sizes are calculated as $K \cdot |c| = 32\ K$ bits for S_1 and $4|H| + |p| = 320$ bits per u_k.

7.2. TD-RCRF Performance

7.2.1. Computational Overhead

Initialization phase. Mobile user u_k first computes a reputation commitment using two exponentiations and one multiplication, incurring a cost of $2T_e + T_m$. To assist the TA in verifying the integrity of its reputation value, u_k then performs an additional four exponentiations, two multiplications, and one hash operation, contributing $4T_e + 2T_m + T_{hg}$ to the total. Following this, they apply secret sharing to divide sensing data and a multiplication triple into two parts, consuming approximately 0.1 ms. In total, u_k's computational overhead in this phase is $4T_e + 2T_m + T_{hg} + 0.1$ ms. Cloud server S_1 authenticates K users by executing K hash generations, with a total cost of $K \cdot T_{hg}$. As S_2 does not participate in this phase, it incurs no computational load.

Truth discovery phase. For the mobile user u_k, the user generates two multiplicative triples and divides the triples and the sensory data into two shares by means of the additive secret sharing scheme $SS(\cdot)$ and sends them to the two servers, respectively, with a computational overhead of 1.6 ms.

For the cloud servers, due to the use of the additive secret sharing technique, the operations completed on the two servers are the same, so only computing one server's computational overhead is sufficient. In the weight update phase, S_1 needs to call the AddSecRes algorithm three times and the MulSecRes algorithm once, and after one multiplication operation, the computational overhead is $3T_{add} + T_{mul} + T_m$. In the truth update phase, server S_1 needs to call the AddSecRes algorithm three times and MulSecRes once, and then perform one multiplication operation and one multiplication ternary operation, with a computational overhead of $3T_{add} + T_{mul} + T_m + T_{tri}$.

To enhance the credibility of the experimental results, Figure 3 illustrates the runtime comparison for mobile user u_k and cloud servers (S_1, S_2) under TD-RCRF, PPTD, and RPTD. As shown, increasing K leads to higher execution times for both users and servers. When $K = 10$, the runtimes of u_k are 16 ms, 900.6 ms, and 1650.4 ms under TD-RCRF, PPTD, and RPTD, respectively, indicating TD-RCRF significantly reduces the user's computational delay. For cloud servers, the corresponding runtimes are 366.2 ms (TD-RCRF), 200.9 ms (PPTD), and 1650.4 ms (RPTD). The improved efficiency of TD-RCRF stems from offloading more computation to the cloud, which typically has a greater processing capacity than mobile clients.

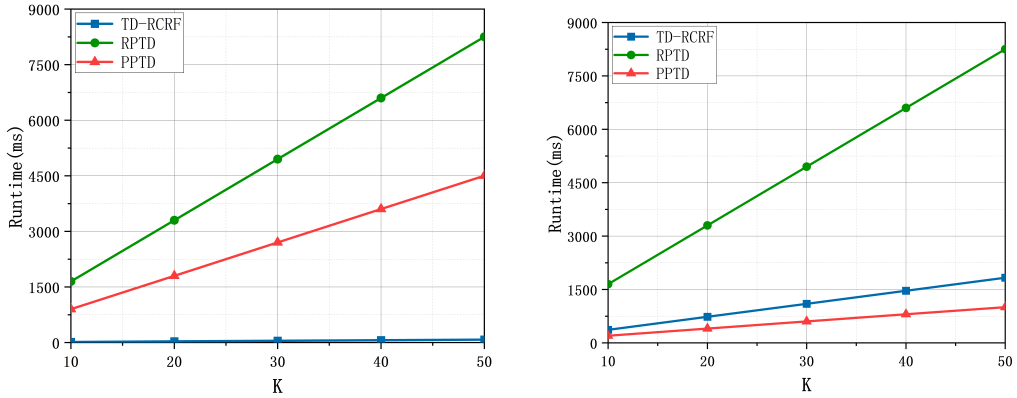


Figure 3. Running overhead associated with (**a**) mobile user u_k and (**b**) cloud servers (S_1, S_2) as K increases incrementally from 10 to 50.

7.2.2. Communication Overhead

Initialization phase: Mobile user u_k transmits the task set $\{T_m\}_{m=1}^{M}$ to S_1, requiring $M \cdot |c|$ bits. Subsequently, they submit the set $\{H_k, D_k, D_{k_1}, D_{k_2}, Cm'_k\}$, contributing an additional $4|H| + |p|$ bits. Hence, the user's total communication burden during this phase equals $M \cdot |c| + 4|H| + |p|$. As for cloud server S_1, it first broadcasts $\{T_m\}_{m=1}^{M}$ to users, and then forwards relevant information to the TA, amounting to a total transmission load of $(M + K) \cdot |c|$. Server S_2, which dispatches a duplicate share of $\{T_m\}_{m=1}^{M}$, incurs a communication cost of $M \cdot |c|$ in this stage.

Truth discovery phase. The mobile user u_k transmits two multiplicative triples, $Tri_1 = (a_1, b_1, c_1)$ and $Tri_2 = (a_2, b_2, c_2)$, to cloud servers S_1 and S_2. Given that each element in a triple is 64 bits, the total size per triple is 192 bits, yielding a communication cost of 0.0469 KB for u_k. During the weight update phase, the servers exchange aggregated values once, incurring 0.0624 KB of overhead. In the truth update phase, the servers perform four rounds of sum exchange and share $\{d\}$ and $\{e\}$, resulting in a cost of 0.078 KB. Thus, the cumulative communication burden on S_1 and S_2 throughout the truth discovery stage amounts to 0.1404 KB.

To strengthen the credibility of our experimental findings, Figure 4 presents a comparative analysis of the communication costs incurred by mobile user u_k and cloud servers

(S_1, S_2) across TD-RCRF, PPTD, and RPTD. As K increases, both user-side and server-side overheads exhibit a rising trend. Notably, TD-RCRF shows a clear advantage in user communication cost compared to the other schemes. For instance, when $K = 10$, the overhead for u_k in TD-RCRF, PPTD, and RPTD is 0.469 KB, 1.875 KB, and 0.98 KB, respectively. Similarly, the server communication costs for TD-RCRF, PPTD, and RPTD are 1.404 KB, 1.29 KB, and 48.5 KB, respectively. These results demonstrate that TD-RCRF achieves significantly lower communication overhead for both users and cloud entities compared to PPTD and RPTD.

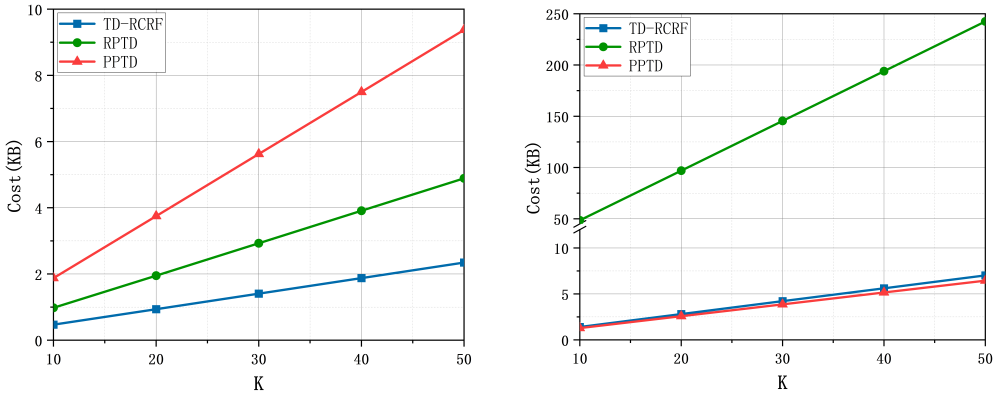


Figure 4. Communication overhead associated with (**a**) mobile user u_k and (**b**) cloud servers (S_1, S_2) as K increases incrementally from 10 to 50.

7.3. Performance of Truth Discovery in TD-RCRF

7.3.1. Weighting of Mobile Users

The TD-RCRF begins by selecting trustworthy workers through a reputation-based filtering mechanism, which enhances the precision of estimating the truth value in sensing tasks. In contrast, classical truth discovery approaches like CRH derive truth estimates directly from raw sensing inputs and user-assigned weights, where these weights are themselves inferred from the sensing data. Since both TD-RCRF and CRH perform truth estimation, we analyze and contrast their respective user weighting mechanisms. As illustrated in Figure 5, when a participant fails to meet the predefined reputation threshold, their assigned weight in TD-RCRF is reduced to zero, and the user is excluded from further computation. Here, $K = 50, 1 \leqslant k \leqslant K$.

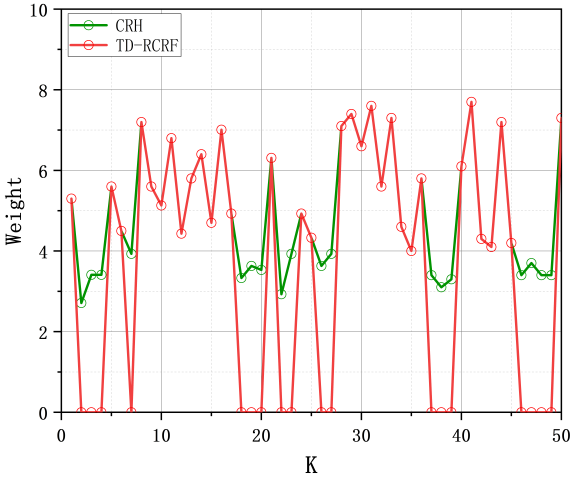


Figure 5. User weights in TD-RCRF and CRH.

7.3.2. Efficiency Assessment

This study utilizes synthetic datasets for experimentation. In particular, the sensing readings from clients follow a normal distribution. The parameters for the count of workers or sensing tasks are chosen to be comparable in scale to those used in CRH [33] and previous research. These parameters reflect real-world crowdsourced sensing deployments like urban environmental monitoring (e.g., air quality/noise), where participant counts typically range in the tens. Our evaluation benchmarks truth discovery (TD) efficiency through computational overhead and accuracy against CRH. For statistical significance, 500 comparative TD sessions were executed for TD-RCRF and CRH, with their mean outcomes illustrated in Figures 6 and 7.

As illustrated in Figure 6, the runtimes of both TD-RCRF and CRH decrease as ϵ increases, with $K = 10$. TD-RCRF incurs a slightly longer runtime than CRH, mainly due to its integration of mobile users' reputation factors, though both remain within the same order of magnitude. In particular, TD-RCRF and CRH require 0.246 s and 0.214 s, respectively, indicating a 13% improvement by TD-RCRF when $\epsilon = 0.01$.

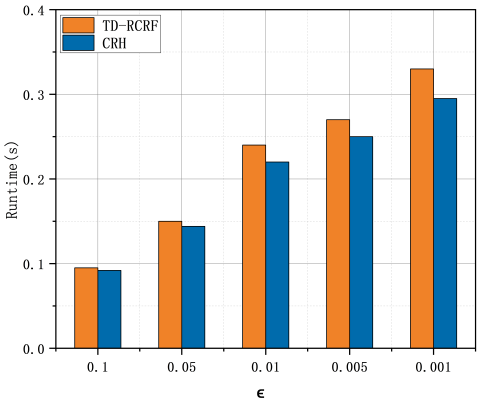


Figure 6. Comparison of running time with different $\epsilon(K = 10)$.

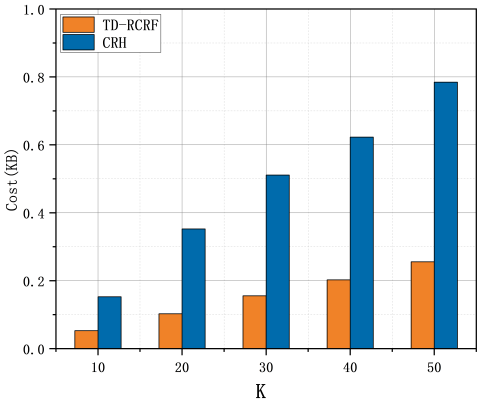


Figure 7. Comparison of cost with different $K(\epsilon = 0.01)$ for the same sensing task.

Figure 7 shows that computational cost rises with increasing K in both schemes, under the assumption that sensory data has a bit-length of 32 and $\epsilon = 0.01$. Notably, CRH aggregates more user data since it considers the total number of subscribers linked to the K mobile users, rather than a direct count of K users. This leads to TD-RCRF consistently incurring lower communication overhead than CRH. For example, when $\epsilon = 0.01$, the communication cost for TD-RCRF is 0.109 KB compared to 0.338 KB in CRH, representing a 67.8% reduction.

Accuracy MAE. From Figure 8, it can be seen that in TD-RCRF and CRH, when ϵ is certain and K increases, the MAE decreases gradually. And with the change in K, the MAE of TD-RCRF is always smaller or equal to the MAE of CRH.

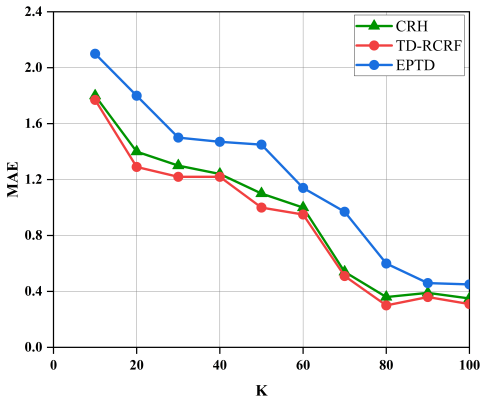


Figure 8. Comparison of MAE for different $K(\epsilon = 0.01)$ on the same sensing task.

Accuracy RMSE. From Figure 9, it can be seen that in TD-RCRF and CRH, when ϵ is constant and K increases, the RMSE decreases gradually. And, with the change in K, the RMSE of TD-RCRF is always smaller than the RMSE of CRH.

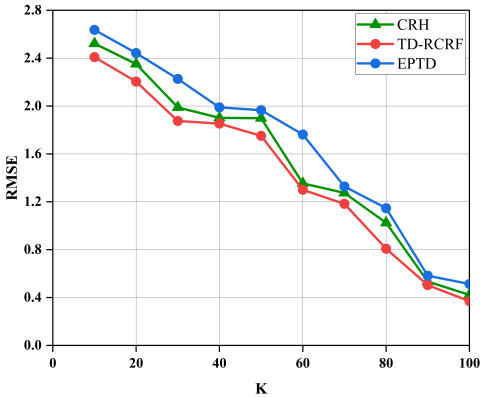


Figure 9. Comparison of RMSE for different $K(\epsilon = 0.01)$ in the same sensing task.

8. Conclusions

In this paper, we propose a privacy-preserving secret sharing-based truth discovery framework. The scheme can be widely applied to various mobile crowdsensing scenarios. We adopt secret sharing to protect the privacy of sensing data and weights, utilize the Pedersen commitment mechanism to protect the privacy of reputation values, and enhance the accuracy of truth discovery by screening highly reputable users. Meanwhile, the scheme uses a dual-cloud architecture to cooperate with the calculation of truth, reducing the computational and communication burden on the user side. Theoretical analysis and experimental evaluation show that the scheme performs well in terms of privacy protection and computing efficiency, and is a safe and reliable scheme.

In the future, our research will focus on two forward-looking directions. First, we will extend the existing scheme to edge computing environments to reduce the impact of network instability on privacy data aggregation. Second, we will design a lighter privacy protection scheme to reduce the computation and communication costs for mobile users, making it more suitable for resource-constrained devices and applications with high real-time requirements.

Author Contributions: Conceptualization, L.B., L.W. and W.W.; methodology, L.B., L.W. and W.W.; software, L.B. and L.W.; validation, L.B., L.W. and W.W.; formal analysis, L.B. and L.W.; investigation, L.W., W.W. and H.P.; resources, L.B. and L.W.; data curation, L.B. and L.W.; writing—original draft preparation, L.B. and L.W.; writing—review and editing, L.B., L.W., W.W. and H.P.; supervision, L.W. and H.P.; project administration, L.W.; funding acquisition, L.W. and W.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under Grants 62472266, 62502291, and U25B2031, and in part by the Natural Science Foundation of Shandong Province under Grant ZR2025MS1027.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

References

1. Guo, B.; Yu, Z.; Chen, L.; Zhou, X.; Ma, X. MobiGroup: Enabling lifecycle support to social activity organization and suggestion with mobile crowd sensing. *IEEE Trans.-Hum.-Mach. Syst.* **2015**, *46*, 390–402. [CrossRef]
2. Liu, L.; Liu, W.; Zheng, Y.; Ma, H.; Zhang, C. Third-eye: A mobilephone-enabled crowdsensing system for air quality monitoring. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* **2018**, *2*, 1–26. [CrossRef]
3. Huang, H.; Yang, J.; Huang, H.; Song, Y.; Gui, G. Deep learning for super-resolution channel estimation and DOA estimation based massive MIMO system. *IEEE Trans. Veh. Technol.* **2018**, *67*, 8549–8560. [CrossRef]
4. Ouyang, R.W.; Kaplan, L.M.; Toniolo, A.; Srivastava, M.; Norman, T.J. Parallel and streaming truth discovery in large-scale quantitative crowdsourcing. *IEEE Trans. Parallel Distrib. Syst.* **2016**, *27*, 2984–2997. [CrossRef]
5. Yin, X.; Han, J.; Yu, P.S. Truth discovery with multiple conflicting information providers on the web. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, CA, USA, 12–15 August 2007; pp. 1048–1052.
6. Zheng, Y.; Duan, H.; Yuan, X.; Wang, C. Privacy-aware and efficient mobile crowdsensing with truth discovery. *IEEE Trans. Dependable Secur. Comput.* **2017**, *17*, 121–133. [CrossRef]
7. Tang, X.; Wang, C.; Yuan, X.; Wang, Q. Non-interactive privacy-preserving truth discovery in crowd sensing applications. In Proceedings of the IEEE INFOCOM 2018—IEEE Conference on Computer Communications, Honolulu, HI, USA, 15–19 April 2018; pp. 1988–1996.
8. Xu, G.; Li, H.; Liu, S.; Wen, M.; Lu, R. Efficient and privacy-preserving truth discovery in mobile crowd sensing systems. *IEEE Trans. Veh. Technol.* **2019**, *68*, 3854–3865. [CrossRef]
9. Liang, Y.; Li, Y.; Shin, B.S. Privacy-Preserving and Reliable Truth Discovery for Heterogeneous Fog-Based Crowdsensing. *IEEE Trans. Dependable Secur. Comput.* **2025**, *22*, 2338–2351. [CrossRef]
10. Alamri, B.H.S.; Monowar, M.M.; Alshehri, S.; Zafar, M.H. A fog-assisted group-based truth discovery framework over mobile crowdsensing data streams. *PLoS ONE* **2025**, *20*, e0330656. [CrossRef] [PubMed]
11. Zhang, C.; Zhu, L.; Xu, C.; Sharif, K.; Ding, K.; Liu, X.; Du, X.; Guizani, M. TPPR: A trust-based and privacy-preserving platoon recommendation scheme in VANET. *IEEE Trans. Serv. Comput.* **2019**, *15*, 806–818. [CrossRef]
12. Li, Q.; Li, Y.; Gao, J.; Su, L.; Zhao, B.; Demirbas, M.; Fan, W.; Han, J. A confidence-aware approach for truth discovery on long-tail data. *Proc. VLDB Endow.* **2014**, *8*, 425–436. [CrossRef]
13. Rana, R.; Chou, C.T.; Bulusu, N.; Kanhere, S.; Hu, W. Ear-Phone: A context-aware noise mapping using smart phones. *Pervasive Mob. Comput.* **2015**, *17*, 1–22. [CrossRef]
14. Xu, G.; Li, H.; Tan, C.; Liu, D.; Dai, Y.; Yang, K. Achieving efficient and privacy-preserving truth discovery in crowd sensing systems. *Comput. Secur.* **2017**, *69*, 114–126. [CrossRef]
15. Zhou, J.; Cao, Z.; Dong, X.; Lin, X. Security and privacy in cloud-assisted wireless wearable communications: Challenges, solutions, and future directions. *IEEE Wirel. Commun.* **2015**, *22*, 136–144. [CrossRef]
16. Lu, R.; Liang, X.; Li, X.; Lin, X.; Shen, X. EPPA: An efficient and privacy-preserving aggregation scheme for secure smart grid communications. *IEEE Trans. Parallel Distrib. Syst.* **2012**, *23*, 1621–1631. [CrossRef]
17. Li, S.; Xue, K.; Yang, Q.; Hong, P. PPMA: Privacy-preserving multisubset data aggregation in smart grid. *IEEE Trans. Ind. Inform.* **2017**, *14*, 462–471. [CrossRef]
18. Miao, C.; Su, L.; Jiang, W.; Li, Y.; Tian, M. A lightweight privacy-preserving truth discovery framework for mobile crowd sensing systems. In Proceedings of the IEEE INFOCOM 2017—IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017; pp. 1–9.

19. Zheng, Y.; Duan, H.; Wang, C. Learning the truth privately and confidently: Encrypted confidence-aware truth discovery in mobile crowdsensing. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2475–2489. [CrossRef]
20. Hu, H.; Lu, R.; Zhang, Z.; Shao, J. REPLACE: A reliable trust-based platoon service recommendation scheme in VANET. *IEEE Trans. Veh. Technol.* **2016**, *66*, 1786–1797. [CrossRef]
21. Ni, J.; Zhang, K.; Xia, Q.; Lin, X.; Shen, X.S. Enabling strong privacy preservation and accurate task allocation for mobile crowdsensing. *IEEE Trans. Mob. Comput.* **2019**, *19*, 1317–1331. [CrossRef]
22. Liu, Z.; Huang, F.; Weng, J.; Cao, K.; Miao, Y.; Guo, J.; Wu, Y. BTMPP: Balancing trust management and privacy preservation for emergency message dissemination in vehicular networks. *IEEE Internet Things J.* **2020**, *8*, 5386–5407. [CrossRef]
23. Cheng, Y.; Ma, J.; Liu, Z.; Wu, Y.; Wei, K.; Dong, C. A lightweight privacy preservation scheme with efficient reputation management for mobile crowdsensing in vehicular networks. *IEEE Trans. Dependable Secur. Comput.* **2022**, *20*, 1771–1788. [CrossRef]
24. Ma, L.; Liu, X.; Pei, Q.; Xiang, Y. Privacy-preserving reputation management for edge computing enhanced mobile crowdsensing. *IEEE Trans. Serv. Comput.* **2018**, *12*, 786–799. [CrossRef]
25. Yan, L.; Yang, K.; Yang, S. Reputation-based truth discovery with long-term quality of source in Internet of Things. *IEEE Internet Things J.* **2021**, *9*, 5410–5421. [CrossRef]
26. Pedersen, T.P. Non-interactive and information-theoretic secure verifiable secret sharing. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 11–15 August 1991; pp. 129–140.
27. Cai, C.; Zheng, Y.; Du, Y.; Qin, Z.; Wang, C. Towards private, robust, and verifiable crowdsensing systems via public blockchains. *IEEE Trans. Dependable Secur. Comput.* **2019**, *18*, 1893–1907. [CrossRef]
28. Zhao, B.; Tang, S.; Liu, X.; Zhang, X. PACE: Privacy-preserving and quality-aware incentive mechanism for mobile crowdsensing. *IEEE Trans. Mob. Comput.* **2020**, *20*, 1924–1939. [CrossRef]
29. Yang, S.; Wu, F.; Tang, S.; Gao, X.; Yang, B.; Chen, G. On designing data quality-aware truth estimation and surplus sharing method for mobile crowdsensing. *IEEE J. Sel. Areas Commun.* **2017**, *35*, 832–847. [CrossRef]
30. Bogdanov, D.; Laur, S.; Willemson, J. Sharemind: A framework for fast privacy-preserving computations. In Proceedings of the Computer Security-ESORICS 2008: 13th European Symposium on Research in Computer Security, Málaga, Spain, 6–8 October 2008; pp. 192–206.
31. Bogdanov, D.; Niitsoo, M.; Toft, T.; Willemson, J. High-performance secure multi-party computation for data mining applications. *Int. J. Inf. Secur.* **2012**, *11*, 403–418. [CrossRef]
32. Bar-Ilan, J.; Beaver, D. Non-cryptographic fault-tolerant computing in constant number of rounds of interaction. In Proceedings of the Eighth Annual ACM Symposium on Principles of Distributed Computing, Edmonton, AB, Canada, 14–16 August 1989; pp. 201–209.
33. Li, Q.; Li, Y.; Gao, J.; Zhao, B.; Fan, W.; Han, J. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, Snowbird, UT, USA, 22–25 June 2014; pp. 1187–1198.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.