This paper has been accepted by IEEE 40th Annual International Conference on Data Engineering (ICDE2024)

LDPRecover: Recovering Frequencies from Poisoning Attacks against Local Differential Privacy

Xinyue Sun[1,2], Qingqing Ye[2], Haibo Hu[2], Jiawei Duan[2], Tianyu Wo[1], Jie Xu[3], Renyu Yang[1]
[1]Beihang University; [2]The Hong Kong Polytechnic University; [3]The University of Leeds
{xy.sun, woty, renyuyang}@buaa.edu.cn; {qqing.ye, haibo.hu}@polyu.edu.hk, jiawei.duan@connect.polyu.hk; j.xu@leeds.ac.uk

***Abstract*—Local differential privacy (LDP), which enables an untrusted server to collect aggregated statistics from distributed users while protecting the privacy of those users, has been widely deployed in practice. However, LDP protocols for frequency estimation are vulnerable to poisoning attacks, in which an attacker can poison the aggregated frequencies by manipulating the data sent from malicious users. Therefore, it is an open challenge to recover the accurate aggregated frequencies from poisoned ones.**

In this work, we propose *LDPRecover*, a method that can recover accurate aggregated frequencies from poisoning attacks, even if the server does not learn the details of the attacks. In LDPRecover, we establish a genuine frequency estimator that theoretically guides the server to recover the frequencies aggregated from genuine users' data by eliminating the impact of malicious users' data in poisoned frequencies. Since the server has no idea of the attacks, we propose an adaptive attack to unify existing attacks and learn the statistics of the malicious data within this adaptive attack by exploiting the properties of LDP protocols. By taking the estimator and the learning statistics as constraints, we formulate the problem of recovering aggregated frequencies to approach the genuine ones as a *constraint inference* (CI) problem. Consequently, the server can obtain accurate aggregated frequencies by solving this problem optimally. Moreover, LDPRecover can serve as a frequency recovery paradigm that recovers more accurate aggregated frequencies by integrating attack details as new constraints in the CI problem. Our evaluation on two real-world datasets, three LDP protocols, and untargeted and targeted poisoning attacks shows that LDPRecover is both accurate and widely applicable against various poisoning attacks.

I. Introduction

Local differential privacy (LDP) [1], a variant of differential privacy [2], [3], is an emerging paradigm that enables an untrusted server to gather aggregated statistics from distributed users while providing provable privacy protection for these users. In LDP, participating users perturb their data locally and report the perturbed data to the untrusted server. Then the server aggregates the statistics of interest from these perturbed data. Thanks to its rigorous privacy guarantee, LDP has been widely deployed in practice. For example, Google [4]–[6] has integrated LDP in Chrome to collect default homepages and search engines; Apple [7] gathers popular emojis and words by deploying LDP in IOS.

However, due to its distributed settings, LDP is vulnerable to *poisoning attacks* [8]–[10], where an attacker may hijack users or inject malicious users to corrupt the LDP protocol. For example, the attacker poisons the aggregated frequencies of arbitrary items by crafting the data sent to the server from these malicious users, as shown in Figure 1. Poisoning attacks against LDP protocols can be further divided into *untargeted* [8] and *targeted poisoning attacks* [9], [10]. In the untargeted attacks, the attacker aims to degrade the overall accuracy of the aggregated frequencies for all items. In the targeted attacks, on the other hand, the attacker wants to increase the aggregated frequencies of attacker-chosen items (i.e., target items) and thus promote them as popular items. Regardless of poisoning types, the server needs to recover accurate aggregated frequencies from the poisoned ones.

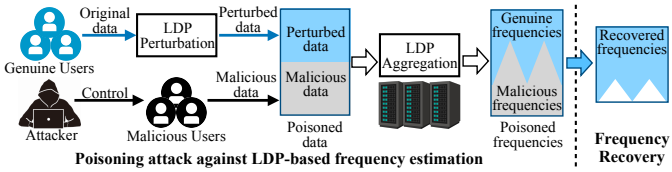

Fig. 1. Illustration of poisoning attack against LDP-based frequency estimation and our frequency recovery.

In the literature, frequency recovery in the LDP protocol suffering from poisoning attacks is largely unexplored. Although there are a few countermeasures [9] against targeted attacks, such as detecting malicious users, they are only effective on specific attacks (e.g., MGA [9]) and require the server to know the details of these attacks.

Our Contributions. In this work, we propose *LDPRecover*, a method that can recover accurate aggregated frequencies from the poisoned ones under LDP protocols, even if the details of the attacks are unknown to the server. While LDPRecover works against both untargeted and targeted attacks by enhancing the overall accuracy of items' aggregated frequencies, for the targeted attack (e.g., MGA [9]), LDPRecover can also reduce the frequency gains (i.e., the increase in frequency) by the attacker on the target items.

LDPRecover is built on our insight that in a poisoning attack, the *poisoned frequencies* aggregated by the server are mixture of *genuine frequencies* aggregated from genuine users' data and *malicious frequencies* aggregated from malicious users' data, as shown in Figure 1. As such, the server can recover genuine frequencies by deducting malicious frequencies from the poisoned frequencies. However, this idea poses non-trivial challenges. First, the theoretical relationship

Qingqing Ye and Tianyu Wo are the corresponding authors.

between the distributions of poisoned, genuine, and malicious frequencies is unexplored. Second, even if this relationship could be derived, the server has yet no prior information about malicious users to recover the genuine frequencies.

To address these challenges, we first propose an analytical framework that generalizes the poisoning attacks against LDP protocols, from which we derive the theoretical relationship between poisoned, genuine, and malicious frequencies. On this basis, we establish a genuine frequency estimator that guides the server to recover the genuine frequencies from the poisoned ones. Then, we propose an adaptive attack to unify state-of-the-art untargeted and targeted attacks, in which we can learn the statistics of malicious frequencies by leveraging the aggregated properties of LDP protocols. By taking both the genuine frequency estimator and the learnt malicious statistics as constraints, we formulate the problem of recovering aggregated frequencies to approach the genuine ones as an *constraint inference (CI) problem*, whose objective function is to minimize the L_2 norm between the recovered and genuine frequencies. Thus, LDPRecover can recover the aggregated frequencies with high accuracy by solving the CI problem optimally.

While LDPRecover does not depend on any specific details about the attacks, these details can help LDPRecover to recover more accurate aggregated frequencies by integrating them as new constraints in the CI problem. For example, when a targeted attack causes a significant increase in target items' frequencies, conventional outlier detection techniques [11]–[13] can identify these target items by detecting statistical anomalies in the historical frequency data of each item. In this case, LDPRecover can exploit such knowledge of target items to recover more accurate aggregated frequencies.

We empirically evaluate our proposed LDPRecover using two datasets, three popular LDP protocols (i.e., GRR [14], OUE, and OLH [15]), as well as three poisoning attacks to LDP (i.e., an untargeted poisoning attack called Manip [8], a targeted poisoning attack called MGA [9], and our proposed adaptive attack called AA). Results show that LDPRecover not only recovers accurate aggregated frequencies from the poisoned ones but also substantially reduces the frequency gains of the targeted poisoning attacks. Our contributions can be summarized as follows:

- To the best of our knowledge, we are the first to put forward a systematic study on frequency recovery from LDP protocols that suffer from general poisoning attacks.
- We propose an LDPRecover method to recover the accurate aggregated frequencies from poisoned ones, even if the server does not learn the details of the attacks. Furthermore, LDPRecover can serve as a frequency recovery paradigm that further improves the accuracy of the recovered frequencies by integrating attack details.
- We evaluate the effectiveness of LDPRecover for three popular LDP protocols suffering from poisoning attacks on two real-world datasets. Results reveal that our proposed method can effectively recover accurate aggregated frequencies and counter these poisoning attacks.

The rest of the paper is organized as follows. We discuss related work in Section II, provide the preliminaries in Section III, and formulate our problem statement in Section IV. Then, we present our recovery method called LDPRecover in Section V and show experimental results in Section VI. Finally, we further discuss the applicability of LDPRecover in Section VII and conclude this paper in Section VIII.

II. Related work

Local Differential Privacy. Local differential privacy (LDP) [1]–[3] has become a *de facto* standard privacy model in sensitive data collection and analysis. In LDP, the server, which aims to collect private data from users, is considered untrusted. Each user first locally perturbs her data by a certain LDP mechanism (e.g., GRR [14]) before sending it to an untrusted server. Based on the perturbed data from users, the server can derive certain aggregated statistics without seeing the actual private data of each user. Due to its rigorous privacy properties, LDP has been widely studied in various tasks, including frequency estimation [14]–[22], mean estimation [23]–[27], heavy hitters identification [28]–[31], range queries [32]–[34], and other more complex tasks [35]–[45].

Poisoning Attack against LDP. LDP is vulnerable to poisoning attacks [8]–[10], in which an attacker poisons the server's aggregated statistics by manipulating the data sent from malicious users. Depending on the attackers' goal, poisoning attacks against LDP protocols be categorized into untargeted poisoning attacks [8] and targeted poisoning attacks [9], [10]. In this work, we focus on countering poisoning attacks against LDP protocols for frequency estimation, as these protocols can serve as the building block of more advanced tasks. Thus, we review Manip [8], which is a popular untargeted poisoning attack, and MGA [9], which is a popular targeted poisoning attack. Specifically, Manip seeks to distort the distribution of aggregated frequencies in L_1 norm. Conversely, MGA strives to amplify the frequency gains of items chosen by the attacker, where "frequency gain" denotes the increase in the target item's frequency after the targeted attack. Note that in both attacks, the attacker requires that malicious users directly send the attacker-crafted data to the server, as this is a more effective way in terms of attack results [8], [9].

Countermeasures against Poisoning Attacks to LDP. We note that several countermeasures were proposed to counter the targeted poisoning attack (i.e., MGA), including malicious users detection and conditional probability based detection [9]. Specifically, these countermeasures, which only apply to OUE and OLH, are built on strong assumptions, e.g., the server knows the details of this attack. However, in reality, these assumptions do not always hold, resulting in these countermeasures being invalid in most cases. As for the untargeted poisoning attack, there is no study to deal with it yet.

III. Preliminaries

A. Local Differential Privacy

In LDP [1], there are many users and one server. Each user possesses an item (data) v from a domain D, and the

server, which is not trusted by users, wants to learn statistics among all users' data. To protect privacy, each user perturbs her input item $v \in D$ locally with an algorithm $\Psi(\cdot)$ and sends the perturbed data $\Psi(v)$ to the server. Formally, the privacy requirement is that $\Psi(\cdot)$ satisfies the following property.

Definition 1 (ϵ-Local Differential Privacy [1])**.** *A randomized algorithm $\Psi(\cdot)$ satisfies ϵ-LDP, where $\epsilon >= 0$, if and only if for any two input $v_1, v_2 \in D$, we have*

$$\forall T \subseteq Range(\Psi): \ \Pr[\Psi(v_1) \in T] \leq e^{\epsilon} \Pr[\Psi(v_2) \in T], \quad (1)$$

where $Range(\Psi)$ denotes the set of all possible outputs of Ψ.

The offered privacy is controlled by privacy budget ϵ, i.e., a smaller (resp. larger) ϵ implies a stronger (resp. weaker) privacy level.

B. LDP Protocols for Frequency Estimation

We review three state-of-the-art pure LDP protocols for frequency estimation. These protocols can be specified by a pair of algorithms (Ψ, Φ): each user uses Ψ to perturb her input item, and the server uses Φ to aggregate the items' frequencies in the perturbed data sent from the users.

General Randomized Response (GRR). General Randomized Response (GRR) [14], a generalized version of randomized response [16], is a basic protocol in LDP. In GRR, each user sends her true item $v \in D$ to the untrusted server with probability p or sends a random $v' \neq v$ with probability q. Formally, the perturbation algorithm $\Psi_{\text{GRR}(\epsilon)}$ is defined as

$$\Pr[\Psi_{\text{GRR}(\epsilon)}(v) = b] = \begin{cases} \frac{e^{\epsilon}}{d-1+e^{\epsilon}} \triangleq p, & \text{if } b = v, \\ \frac{1}{d-1+e^{\epsilon}} \triangleq q, & \text{if } b \neq v, \end{cases} \quad (2)$$

where d is the size of D, i.e., $d = |D|$. It is easy to prove this satisfies ϵ-LDP since $\frac{p}{q} = e^{\epsilon}$. To estimate the frequency of $v \in D$, the server first counts v, denoted by $C(v)$, then computes the estimated count of the users who have v as private item:

$$\Phi_{\text{GRR}(\epsilon)}(v) := \frac{C(v) - nq}{p - q}, \quad (3)$$

where n is the total number of users. Then the estimated frequency of v is $\tilde{f}(v) = \frac{1}{n}\Phi_{\text{GRR}(\epsilon)}(v)$. In [15], it is shown that $\Phi_{\text{GRR}(\epsilon)}(\cdot)$ is an unbiased estimation of true counts, and the variance of this estimation is

$$\mathbf{Var}[\Phi_{\text{GRR}(\epsilon)}(v)] = n \cdot \frac{d - 2 + e^{\epsilon}}{(e^{\epsilon} - 1)^2} + nf(v) \cdot \frac{d-2}{e^{\epsilon} - 1} \quad (4)$$

Optimized Unary Encoding (OUE). Optimized Unary Encoding (OUE) protocol is designed to avoid the variance of the estimation depending on the domain size d by encoding the item into the unary representation. In OUE, each user first encodes her item $v \in D$ to a d-bit binary vector $\mathbf{b} = [b_1\ b_2\ \cdots\ b_d]$ whose bits are all 0 except for the 1 in the v-th bit. Then, each user perturbs each bit of her encoded binary vector with $\Phi_{\text{OUE}(\epsilon)}(\cdot)$ independently. Specifically,

$$\Pr[\Psi_{\text{OUE}(\epsilon)}(b_i) = 1] = \begin{cases} \frac{1}{2} \triangleq p, & \text{if } i = v, \\ \frac{1}{e^{\epsilon}+1} \triangleq q, & \text{otherwise}, \end{cases} \quad (5)$$

where $\tilde{b}_i = \Psi_{\text{OUE}(\epsilon)}(b_i)$ is the i-th perturbed bit, and $\tilde{\mathbf{b}} = [\tilde{b}_1\ \tilde{b}_2\ \cdots\ \tilde{b}_d]$ is the perturbed vector.

Given the reports $\tilde{\mathbf{b}}^j$ from all users $j \in [n]$, to estimate the frequency of v, the server counts the number of reports whose v-th bit is set to 1, denoted by $C(v) = |\{j | \tilde{b}_v^j = 1\}|$. Then, the server transforms $C(v)$ to its unbiased estimation by

$$\Phi_{\text{OUE}(\epsilon)}(v) := \frac{C(v) - nq}{p - q}. \quad (6)$$

It is proven in [15] that $\Phi_{\text{OUE}(\epsilon)}(\cdot)$ satisfies ϵ-LDP, and the estimated count is unbiased and has variance

$$\mathbf{Var}[\Phi_{\text{OUE}(\epsilon)}(v)] = n \cdot \frac{4e^{\epsilon}}{(e^{\epsilon} - 1)^2} \quad (7)$$

Optimized Local Hashing (OLH). Optimized Local Hashing (OLH) [15], [21] protocol aims to deal with a large domain size d by applying a hash function to map an input item into a smaller domain of size g (i.e., $g \ll d$). In particular, OLH sets g to $\lceil e^{\epsilon} + 1 \rceil$ by default, as it achieves the lowest variance with this setting. In OLH, each user first randomly picks a hash function H from a family of hash functions $\mathbf{H}$ (e.g., xxhash) and then computes the hashed value of her item v as $b = H(v)$, where $b \in \{1, 2, ..., g\}$ is a value hashed from $v \in D$ using H, and the tuple (H, b) is the encoded value for v. Note that $\mathbf{H}$ should have the property that the distribution of each v's hash value is uniform over $\{1, 2, ..., g\}$ and independent from the distributions of other input items in D. Next, each user perturbs her encoded value by the following perturbation function.

$$\Psi_{\text{OLH}(\epsilon)}(v) := \left\langle H, \Psi_{\text{GRR}(\epsilon)}(H(v)) \right\rangle. \quad (8)$$

where $\Psi_{\text{GRR}(\epsilon)}(\cdot)$ is the perturbation algorithm of GRR on the domain $\{1, 2, ..., g\}$.

Let $\left\langle H^j, b^j \right\rangle$ be the report from the j-th user. To estimate the frequency of $v \in D$, the server first counts the number of reports whose input item could be v, denoted by $C(v) = \{j | H^j(x) = a^j\}$. Then, the server transforms $C(v)$ to its unbiased estimation

$$\Phi_{\text{OLH}(\epsilon)}(v) := \frac{C(v) - nq}{p - q}, \quad (9)$$

where $p = e^{\epsilon}/(e^{\epsilon} + g - 1)$ and $q = 1/g$. The variance of this estimation is

$$\mathbf{Var}[\Phi_{\text{OLH}(\epsilon)}(v)] = n \cdot \frac{4e^{\epsilon}}{(e^{\epsilon} - 1)^2}. \quad (10)$$

C. Summary of Common Properties of LDP Protocols

Here we summarize the common properties of the pure LDP protocols. When the server aggregates the reports from all users, for each item $v \in D$, its estimated count for any LDP protocol can be represented in a unified way:

$$\Phi_{\epsilon}(v) := \frac{C(v) - nq}{p - q}. \quad (11)$$

Note that the perturbed probabilities p and q in various protocols are different. Besides, since these protocols are pure LDP protocols [15], $C(v)$ can be represented as follows.

$$C(v) = \sum_{i=1}^{n} \mathbb{1}_{S(\tilde{v}_i)}(v) \tag{12}$$

where $\mathbb{1}_{S(\tilde{v}_i)}(v)$ is a characteristic function:

$$\mathbb{1}_{S(\tilde{v}_i)}(v) = \begin{cases} 1, & \text{if } v \in S(\tilde{v}_i), \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

In particular, $\tilde{v}_i$ denotes the perturbed data of the i-th user, and $S(\tilde{v}_i)$ denotes the set of items that $\tilde{v}_i$ supports, i.e., the support $S(\tilde{v}_i)$ of a perturbed item $\tilde{v}_i$ is the set of items whose encoded values could be $\tilde{v}_i$.

IV. Problem Definition

A. Threat Model

We focus on the threat model in prior studies of poisoning attacks against LDP protocols [8], [9]. In what follows, we discuss the attacker's goals, capabilities, and background knowledge in detail.

Attacker's goals. In an untargeted poisoning attack, the attacker's goal is to indiscriminately increase the error in the frequencies of items aggregated by the server. In a targeted poisoning attack, the attacker aims to increase the frequencies of the target items chosen by the attacker.

Attacker's Capabilities and Background Knowledge. We assume an attacker can control some malicious users in an LDP protocol. These malicious users could be fake users injected into the LDP protocol or genuine users compromised by the attacker. The attacker crafts the malicious data sent from these malicious users to the server.

Since the LDP protocol is executed on the users' side, the attacker knows the details of the LDP protocol adopted by the genuine users. Specifically, the attacker knows various parameters of the LDP protocols, including input domain D, encoded domain $\tilde{D}$, and privacy budget ϵ.

B. Design Goals

We aim to design an accurate and widely-applicable frequency recovery method for LDP protocols that suffer from poisoning attacks. Even without knowledge about the poisoning attacks, our recovery method should still be able to recover aggregated frequencies close to the genuine ones. Specifically, our design goals are as follows.

Accuracy. The aggregated frequencies recovered by our recovery method should be accurate. For both untargeted and targeted poisoning attacks, the recovered frequencies should be close to the genuine frequencies that the server gathers from genuine users using an LDP protocol. Furthermore, for the targeted poisoning attack, we require that the frequency gains of that attack in the aggregated frequencies recovered by our method should be very low.

Applicability. Our recovery method should be widely applicable to counter the poisoning attacks to LDP protocols, even if the server does not know the details of the attacks. In particular, the server may acquire partial knowledge about the attacks in some applications. For example, the server, utilizing outlier detection methodologies [11]–[13], can deduce the attacker's target items through careful analysis of historical data. In this case, our recovery method should be able to recover more accurate frequencies by exploiting such knowledge of these target items.

V. LDPRecover

A. Overview

LDPRecover is based on the following insights. Note that in a poisoning attack, the poisoned frequencies are mixture of genuine and malicious frequencies. Suppose we have a genuine frequency estimator that enables the server to recover the genuine frequencies by deducting the malicious frequencies from the poisoned ones. Then, if we can learn the malicious frequencies of items, the server can recover the aggregated frequencies close to the genuine ones from the poisoned ones. Following the insights, we design LDPRecover with three major parts: estimator construction, malicious frequency learning, and genuine frequency recovery.

Step 1: Estimator Construction. The first step of LDPRecover is to construct a genuine frequency estimator, which guides the server on how to recover the genuine frequencies. As such, we first propose an analytical framework to generalize the poisoning attacks against LDP protocols, by which we further derive the theoretical relationship between poisoned, genuine, and malicious frequencies. On this basis, we establish the genuine frequency estimator and analyze the expectation and variance of the estimator. We provide the details of this step in Section V-B and further give the error analysis of this estimator in Section V-E.

Step 2: Malicious Frequency Learning. Since we assume that the server has no details of the poisoning attacks, we cannot obtain the malicious frequencies of items directly. To tackle this challenge, we propose an adaptive attack that unifies state-of-the-art untargeted and targeted attacks [8], [9]. Following this, we leverage the aggregated properties of LDP protocols to learn the statistics of malicious frequencies, specifically their summation, within the adaptive attack. These learning statistics serve as an alternative approximation for the malicious frequencies. We give the details of this part in Section V-C.

Step 3: Genuine Frequency Recovery. By treating the genuine frequency estimator and the learning statistics of malicious frequencies as constraints, we formulate the problem of recovering aggregated frequencies to approach the genuine ones as a Constraint Inference (CI) problem. The server can obtain accurate aggregated frequencies by solving this CI problem. Note that both the genuine frequency estimator and the statistics of the malicious frequencies are derived from the public information known to the server, such as the LDP protocols. Therefore, LDPRecover can work even if the server has no details of the attacks. More importantly, when the

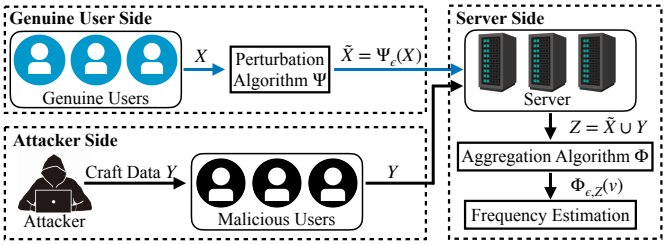

Fig. 2. The general process of poisoning attacks against LDP protocols.

server is able to acquire partial knowledge about the attacks, LDPRecover can also exploit such knowledge to help the server better counter the attacks and improve the accuracy of the recovered frequencies. More details of this step are illustrated in Section V-D.

B. Estimator Construction

This subsection corresponds to Step 1 of LDPRecover. Here we present a general framework for poisoning attacks against LDP protocols, by which we further establish the genuine frequency estimator.

1) Analytical Framework for Poisoning Attacks: Our framework includes three parties: genuine users, an attacker, and a server. Figure 2 summarizes our framework.

- **Genuine User Side:** Suppose there are n genuine users. Each user possesses a private item x and perturbs it into $\tilde{x} = \Psi_\epsilon(x)$ using the perturbation algorithm $\Psi_\epsilon(\cdot)$ of an LDP protocol with a privacy budget ϵ. Then, the genuine users report the perturbed data to the server. In particular, $\tilde{x}$ could be an index (e.g., for GRR), a binary vector (e.g., for OUE), and a tuple (e.g., for OLH). We use $x_i \in D$ and $\tilde{x}_i \in \tilde{D}$ to denote the original item and perturbed data of the i-th user, where D and $\tilde{D}$ denote the input domain and encoded domain of the LDP protocol, respectively. Moreover, we use $X = \{x_1, x_2, ..., x_n\}$ and $\tilde{X} = \{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n\}$ to denote the original items and perturbed data of n genuine users, respectively.
- **Attacker Side:** The attacker crafts data in $\tilde{D}$ for m malicious users, and these users directly send the attacker-crafted data to the server. We use $Y = \{y_1, y_2, ..., y_m\}$ to denote the attacker-crafted data sent from these malicious users.
- **Server Side:** Once receiving the reported data from genuine and malicious users, the server could estimate the count of each item $v \in D$ from the reported data. Let $Z = \{z_1, z_2, ..., z_{n+m}\}$ denote the reported data, i.e., $Z = \tilde{X} \cup Y$, and $\Phi_{\epsilon,Z}(v)$ be the estimated count of v in Z, where $\Phi_\epsilon(\cdot)$ denotes the aggregation algorithm executed on Z.

Under this framework, we can analyze the theoretical relationship between the frequency of each item in $\tilde{X}$, Y, and Z. Specifically, for each item $v \in D$, we use $\tilde{f}_{\tilde{X}}(v)$, $\tilde{f}_Y(v)$ and $\tilde{f}_Z(v)$ to denote the *genuine*, *malicious*, and *poisoned frequency* of v, where $\tilde{f}_{\tilde{X}}(v)$, $\tilde{f}_Y(v)$ and $\tilde{f}_Z(v)$ are **aggregated** from $\tilde{X}$, Y, and Z, respectively, using an LDP protocol. The relationship between $\tilde{f}_{\tilde{X}}(v)$, $\tilde{f}_Y(v)$ and $\tilde{f}_Z(v)$ can be represented as follows.

$$\tilde{f}_Z(v) = \frac{n}{n+m}\tilde{f}_{\tilde{X}}(v) + \frac{m}{n+m}\tilde{f}_Y(v). \tag{14}$$

This equation implies that $\tilde{f}_Z(v)$ can be decomposed into a linear combination of $\tilde{f}_{\tilde{X}}(v)$ and $\tilde{f}_Y(v)$. However, due to the randomness of LDP protocols, $\tilde{f}_{\tilde{X}}(v)$ and $\tilde{f}_Y(v)$ are distributions but not deterministic values. Therefore, we need to establish the relationship of the distributions of $\tilde{f}_{\tilde{X}}(v)$, $\tilde{f}_Y(v)$, and $\tilde{f}_Z(v)$. To be specific, for the data sent from genuine users and malicious users (i.e., $\tilde{X}$ and Y), we use $\Phi_{\epsilon,\tilde{X}}(v)$ and $\Phi_{\epsilon,Y}(v)$ to denote the estimated counts of v in $\tilde{X}$ and Y, respectively. Consequently, $\tilde{f}_{\tilde{X}}(v)$ and $\tilde{f}_Y(v)$ can be expressed as:

$$\tilde{f}_{\tilde{X}}(v) = \frac{1}{n}\Phi_{\epsilon,\tilde{X}}(v), \ \tilde{f}_Y(v) = \frac{1}{m}\Phi_{\epsilon,Y}(v). \tag{15}$$

Recalling the properties of LDP protocols mentioned in Section III-C, both $\Phi_{\epsilon,\tilde{X}}(v)$ and $\Phi_{\epsilon,Y}(v)$ can be decomposed as follows.

$$\Phi_{\epsilon,\tilde{X}}(v) = \sum_{\tilde{x}\in\tilde{X}} \frac{\mathbb{1}_{S(\tilde{x})}(v) - q}{p-q}, \ \Phi_{\epsilon,Y}(v) = \sum_{y\in Y} \frac{\mathbb{1}_{S(y)}(v) - q}{p-q}, \tag{16}$$

where p and q represent the perturbation probabilities of the LDP protocol. Without loss of generality, we slightly abuse $\Phi_{\epsilon,\tilde{x}}(v)$ and $\Phi_{\epsilon,y}(v)$ to denote the estimated count of v in any data $\tilde{x}$ and y, respectively, as follows:

$$\Phi_{\epsilon,\tilde{x}}(v) = \frac{\mathbb{1}_{S(\tilde{x})}(v) - q}{p-q}, \quad \Phi_{\epsilon,y}(v) = \frac{\mathbb{1}_{S(y)}(v) - q}{p-q}, \tag{17}$$

where $\mathbb{1}_{S(\cdot)}(v)$ is the characteristic function defined in Equation (13). Hence, $\Phi_{\epsilon,\tilde{X}}(v)$ and $\Phi_{\epsilon,Y}(v)$ can be redefined as:

$$\Phi_{\epsilon,\tilde{X}}(v) = \sum\nolimits_{\tilde{x}\in\tilde{X}} \Phi_{\epsilon,\tilde{x}}(v), \Phi_{\epsilon,Y}(v) = \sum\nolimits_{y\in Y} \Phi_{\epsilon,y}(v). \tag{18}$$

Building on this groundwork, we can model the distributions of $\tilde{f}_{\tilde{X}}(v)$ and $\tilde{f}_Y(v)$ using *Lindeberg–Lévy Central Limit Theorem* (CLT) [46], [47].

Model the Distribution of $\tilde{f}_Y(v)$. We first model the distribution of $\tilde{f}_Y(v)$ in the poisoning attack. Note that $\Phi_{\epsilon,Y}(v)$ could be regarded as the summation of m independent identically distributed random variables (i.e., $\Phi_{\epsilon,y}(v)$) [1]. Therefore, the following lemma establishes the asymptotic distribution of $\tilde{f}_Y(v)$.

Lemma 1. *The asymptotic distribution $\tilde{f}_Y(v)$ is $\mathcal{N}(\mu_y, \sigma_y^2)$, $\lim_{m\to\infty} \tilde{f}_Y(v) \sim \mathcal{N}(\mu_y, \sigma_y^2)$, where $\mathcal{N}$ denotes a normal distribution, $\mu_y = \mathbf{E}[\Phi_{\epsilon,y}(v)]$, and $\sigma_y^2 = \mathbf{Var}[\Phi_{\epsilon,y}(v)]/m$.*

Proof. Please refer to our technical report [48]. □

Model the Distribution of $\tilde{f}_{\tilde{X}}(v)$. It is rather challenging to model the distribution of $\tilde{f}_{\tilde{X}}(v)$. Indeed, $\tilde{f}_{\tilde{X}}(v)$ is the

[1]The process of an attacker crafting data for a malicious user is essentially equivalent to sampling from the distribution specified by the attacker (See Section V-C). Thus, the aggregated results of each sample (crafted data) still follow the same distribution, as all samples are executed by the same aggregation algorithm.

estimated frequency of v using the complete algorithm pair (i.e., Ψ and Φ) of the LDP protocol on X. However, each genuine user perturbs her original data x to $\tilde{x} = \Psi_\epsilon(x)$, and different original data follow different perturbations (see Section III-B). Thus, $\{\Phi_{\epsilon,\tilde{x}}(v)|\tilde{x} \in \tilde{X}\}$ are probably not identically distributed, which does not satisfy the prerequisite of *Lindeberg–Lévy* CLT.

Fortunately, it is still viable to model $\tilde{f}_{\tilde{X}}(v)$ as one normal distribution. This is because genuine users with identical input data apply the same perturbation algorithm, leading to independently and identically distributed estimated counts among these users. As such, the perturbed data arising from identical input data can be partitioned into distinct subsets. Following this, one normal distribution can be deployed to approximate the summation of v's aggregated frequencies in each subset. The asymptotic distribution of $\tilde{f}_{\tilde{X}}(v)$ is formalized in the following lemma.

Lemma 2. *The asymptotic distribution of $\tilde{f}_{\tilde{X}}(v)$ is $\mathcal{N}(\mu_{\tilde{x}}, \sigma^2_{\tilde{x}})$, i.e., $\lim_{n\to\infty} \tilde{f}_{\tilde{X}}(v) \sim \mathcal{N}(\mu_{\tilde{x}}, \sigma^2_{\tilde{x}})$, where $\mu_{\tilde{x}} = f_X(v)$, $\sigma^2_{\tilde{x}} = \frac{q(1-q)}{n(p-q)^2} + \frac{f_X(v)(1-p-q)}{n(p-q)}$, and $f_X(v)$ is the true frequency of v in genuine data X.*

Proof. Please refer to our technical report [48]. □

Model the Distribution of $\tilde{f}_Z(v)$. Note that Lemmas 1 and 2 state that the aggregated frequency of each item in the data, sent to the server by either malicious or genuine users, approximates a specific normal distribution. Thus, the distribution of $\tilde{f}_Z(v)$ can be expressed jointly by the distributions of $\tilde{f}_{\tilde{X}}(v)$ and $\tilde{f}_Y(v)$. The following theorem establishes the asymptotic distribution of $\tilde{f}_Z(v)$.

Theorem 1. *The asymptotic distribution of $\tilde{f}_Z(v)$ is $\mathcal{N}(\mu_z, \sigma^2_z)$, i.e., $\lim_{n,m\to\infty} \tilde{f}_Z(v) \sim \mathcal{N}(\mu_z, \sigma^2_z)$, where $\mu_z = \frac{1}{1+\eta}\mu_{\tilde{x}} + \frac{\eta}{1+\eta}\mu_y$, $\sigma^2_z = \frac{1}{(1+\eta)^2}\sigma^2_{\tilde{x}} + \frac{\eta^2}{(1+\eta)^2}\sigma^2_y$, and η denotes the ratio of the number of malicious users to the number of genuine users, i.e., $\eta = \frac{m}{n}$.*

Proof. Please refer to our technical report [48]. □

2) Estimator Construction: Our analytic framework reveals the relationship of the distributions of $\tilde{f}_{\tilde{X}}(v)$, $\tilde{f}_Y(v)$, and $\tilde{f}_Z(v)$. On this basis, we propose the genuine frequency estimator to recover $\tilde{f}_{\tilde{X}}(v)$ from $\tilde{f}_Z(v)$ as follows.

$$\tilde{f}_{\tilde{X}}(v) = (1+\eta)\tilde{f}_Z(v) - \eta\tilde{f}_Y(v). \tag{19}$$

This estimator shows that the server with $\tilde{f}_Z(v)$ can recover $\tilde{f}_{\tilde{X}}(v)$ by removing $\tilde{f}_Y(v)$. Theorems 2 and 3 analyze the expectation and variance of our estimator in Equation (19). As we will show in experiments, setting a larger η is sufficient to ensure good performance of the recovered aggregated frequencies, even though the server does not know the true value of η,

Theorem 2. *The estimator in Equation (19) is approximately unbiased, i.e., $\lim_{n,m\to\infty} \mathbf{E}[\tilde{f}_{\tilde{X}}(v)] = f_X(v)$, where $f_X(v)$ is the true frequency of v in the genuine data X.*

Proof. Please refer to our technical report [48]. □

Theorem 3. *The approximate variance of the estimator (Equation (19)) is $\sigma^2_{\tilde{x}}$, where $\sigma_{\tilde{x}}$ can be obtained from Lemma 2.*

Proof. Please refer to our technical report [48]. □

C. Malicious Frequency Learning

This subsection corresponds to Step 2 in LDPRecover.

Adaptive Attack. Here we introduce an adaptive poisoning attack that unifies existing poisoning attacks [9], [49] on the LDP protocols. We observe that existing poisoning attacks can be characterized as the sampling of malicious data for each malicious user from an attacker-designed distribution. Different attacks utilize different attacker-designed distributions, leading to various attackers' objectives. Accordingly, the adaptive attack is designed as follows: the attacker initially establishes an attacker-designed (adaptive) distribution P over the encoded domain $\tilde{D}$, and subsequently draws samples $v \in \tilde{D}$ from P with probability $P(v)$. These samples are then employed as the crafted data for malicious users. For example, the malicious data in MGA [9] are essentially sampled from the attacker-designed distribution in which the sampling probabilities of all untarget items are 0.

Approximate Summation of Malicious Frequencies. Next, we estimate the statistics of malicious frequencies during an adaptive attack as an approximate substitute for $\tilde{f}_Y(v)$, since the server is unaware of $\tilde{f}_Y(v)$ in the genuine frequency estimator. Specifically, during the adaptive poisoning attack, we can derive the expected summation of malicious frequencies for all items, denoted by $\mathbf{E}[\sum_{v\in D} \tilde{f}_Y(v)]$.

$$\mathbf{E}[\sum\nolimits_{v\in D} \tilde{f}_Y(v)] = \frac{\sum_{v\in D} P(v) - qd}{p-q} = \frac{1-qd}{p-q} \tag{20}$$

where $P(v)$ is the probability of item v in P, d is the size of D, and p, q are perturbation probabilities of the LDP protocol. Note that the value of $\mathbf{E}[\sum_{v\in D} \tilde{f}_Y(v)]$ changes with varying parameters p and q in different LDP protocols. This is due to the fact that the data reported by malicious users bypasses the perturbation algorithm of the LDP protocol, but is subjected to the aggregation algorithm of the LDP protocol. Additionally, regardless of the attacker-designed distribution P, the sum of frequencies across all items always be 1, i.e., $\sum_{v\in D} P(v) = 1$. Consequently, we can use the expected summation to approximate the actual summation of malicious frequencies for all items, i.e.,

$$\sum\nolimits_{v\in D} \tilde{f}_Y(v) \triangleq \mathbf{E}[\sum\nolimits_{v\in D} \tilde{f}_Y(v)] = \frac{1-qd}{p-q}. \tag{21}$$

These approximate summations provide the necessary information for recovering the genuine frequencies from the poisoned ones in the subsequent section.

D. Genuine Frequency Recovery

This subsection corresponds to Step 3 in LDPRecover. Intuitively, we note that the process of recovering genuine frequency is essentially solving a Constraint Inference problem.

A natural solution is to recover genuine frequencies from the poisoned ones using Constrained Least Squares (CLS) [50], where the constraints are the genuine frequency estimator in Step 1 and the estimated summations of malicious frequencies in Step 2. We also impose the publicly prior knowledge that all individual frequencies should be non-negative, and they sum up to one as the constraints to improve the accuracy [21].

Formally, given the poisoned frequency vector $\tilde{\mathbf{f}}_Z$, LDPRecover outputs the recovered frequency vector $\mathbf{f}'_{\tilde{X}}$ by solving the following problem:

$$\begin{aligned} \text{minimize:} \quad & ||\mathbf{f}'_{\tilde{X}} - \tilde{\mathbf{f}}_{\tilde{X}}||_2 \\ \text{subject to:} \quad & \forall_v f'_{\tilde{X}}(v) \geq 0 && (22) \\ & \sum\nolimits_{v\in D} f'_{\tilde{X}}(v) = 1 && (23) \\ & \tilde{f}_{\tilde{X}}(v) = (1+\eta)\tilde{f}_Z(v) - \eta\tilde{f}_Y(v) && (24) \\ & \sum\nolimits_{v\in D} \tilde{f}_Y(v) = \frac{1-qd}{p-q} && (25) \end{aligned}$$

where the first two conditions come from the publicly prior knowledge, and the last two derive from Steps 1 & 2 of LDPRecover. To solve this problem, we first use conditions (24) and (25) to estimate the genuine frequencies $\tilde{\mathbf{f}}_{\tilde{X}}$, and then refine $\mathbf{f}'_{\tilde{X}}$ from $\tilde{\mathbf{f}}_{\tilde{X}}$ under conditions (22) and (23).

Estimating Genuine Frequencies. Specifically, we consider two scenarios: non-knowledge scenario and partial-knowledge scenario. The former assumes that the server is unaware of the attack details, while the latter assumes that the server knows attacker-selected items in the targeted attack.

Non-Knowledge Recovery. In the non-knowledge scenario, the server has no details about the attack. We divide the domain D into two sub-domain $D_0 \subseteq D$ and $D_1 = D \setminus D_0$ such that $D_0 = \{v|\tilde{f}_Z(v) \leq 0\}$ and $D_1 = D \setminus D_0$. Intuitively, the poisoned frequencies of potential items subjected to poisoning attacks should be higher. Thus, the items in D_1 are considered to be potential items subject to poisoning attacks, and their malicious frequencies are assumed as uniform. That is, for any $v \in D_0$, we assign $f'_Y(v) = 0$, while for any $v \in D_1$, we assign

$$\tilde{f}'_Y(v) = \frac{1}{|D_1|}\sum\nolimits_{v\in D} \tilde{f}_Y(v). \tag{26}$$

where $\sum_{v\in D} \tilde{f}_Y(v)$ can be obtained from Equation (25). By replacing $\tilde{f}_Y(v)$ with $\tilde{f}'_Y(v)$ in Equation (24), we obtain the estimated genuine frequency for each $v \in D$ as follow.

$$\tilde{f}_{\tilde{X}}(v) = (1+\eta)\tilde{f}_Z(v) - \eta\tilde{f}'_Y(v) \tag{27}$$

Partial-Knowledge Recovery. We observe that targeted attacks tend to significantly increase the frequencies of attacker-selected items, rendering these items as statistical outliers. There are many outlier detection techniques [11]–[13] available for inferring the target items by analyzing statistical anomalies in the frequency of each item in historical data. For example, these works can encode the historical frequencies of each item as a time series, fit a prediction model to the time series and predict current frequencies using past data. Aggregated frequencies of items that are very different from their predicted frequencies are identified as outliers (i.e., target items). Motivated by this, in the partial-knowledge scenario, we assume that the server has awareness of the items selected by the attacker. In what follows, we illustrate how to integrate the available attack details into LDPRecover to estimate more accurate genuine frequencies.

Specifically, let $\mathcal{T}$ denote the set of attacker-selected items. With $\mathcal{T}$, LDPRecover can update $\tilde{f}'_Y(v)$ in Equation (26) as its more accurate version, denoted by $\tilde{f}^*_Y(v)$. Note that $\mathcal{T}$ affects the estimated result of $\mathbf{f}_Y$. Therefore, we first use $\mathcal{T}$ to obtain $\tilde{f}^*_Y(v)$ for all $v \in D$. In this step, we divide D into two cases: $D' = D \setminus \mathcal{T}, D'' = D \cap \mathcal{T}$. For both cases, we have the following estimated results:

- $v \in D'$: As D' does not include any target item in $\mathcal{T}$, the frequencies of v in D' in malicious data Y are 0, i.e., $P(v) = 0$ for all $v \in D'$. Following Equation (20), the approximate summation of malicious frequencies for all items in D' can be represented as follows.

$$\sum\nolimits_{v\in D'} \tilde{f}_Y(v) \triangleq \frac{\sum_{v\in D'} P(v) - qd}{p-q} = -\frac{qd}{p-q} \tag{28}$$

- $v \in D''$: In this case, v is the attacker-selected item in $\mathcal{T}$. With $\sum_{v\in D'} \tilde{f}_Y(v)$, the approximate summation of malicious frequencies for all items in D'' can be computed as follows.

$$\sum\nolimits_{v\in D''} \tilde{f}_Y(v) = \sum\nolimits_{v\in D} \tilde{f}_Y(v) - \sum\nolimits_{v\in D'} \tilde{f}_Y(v) \tag{29}$$

where $\sum_{v\in D} \tilde{f}_Y(v)$ is obtained from Equation (25).

As we do not know the weights among the attacker-selected items, we again assume that the frequencies of such items are uniformly distributed. Putting things together, we have the estimate the new malicious frequencies $\tilde{f}^*_Y(v)$ as follows.

$$\tilde{f}^*_Y(v) = \begin{cases} -\frac{qd}{|D'|(p-q)}, & v \in D' \\ \frac{1}{|D''|}(\sum\limits_{v\in D} \tilde{f}_Y(v) - \sum\limits_{v\in D'} \tilde{f}_Y(v)), & v \in D'' \end{cases} \tag{30}$$

Next, we can update Equation (27) with $\tilde{f}^*_Y(v)$ to obtain the recovered frequency as follows.

$$\tilde{f}_{\tilde{X}}(v) = (1+\eta)\tilde{f}_Z(v) - \eta\tilde{f}^*_{\tilde{X}}(v) \tag{31}$$

Intuitively, such partial knowledge can improve the accuracy of genuine frequencies, thereby enabling LDPRecover to recover a more accurate aggregated frequency. Our experiments in Section VI-C confirm this intuition.

Refining Recovered Frequencies. With $\tilde{\mathbf{f}}_{\tilde{X}}$, we use the KKT condition [21], [51], [52] to solve this problem under conditions (22) and (23). Specifically, the optimization target can be augmented by the following equations:

$$\begin{aligned} \text{minimize:} \quad & \sum\nolimits_{v\in D}(f'_{\tilde{X}}(v) - \tilde{f}_{\tilde{X}}(v))^2 + a + b \\ \text{where:} \quad & \forall_v f'_{\tilde{X}}(v) \geq 0, \sum\nolimits_v f'_{\tilde{X}}(v) = 1 \\ & a = \mu \sum_v f'_{\tilde{X}}(v) \\ & b = \sum\nolimits_v \lambda_v f'_{\tilde{X}}(v), \forall_v : \lambda_v \times f'_{\tilde{X}}(v) = 0 \end{aligned}$$

Note that $a = \mu$ is a constant, and $b = 0$. Therefore, the conditions of the minimization objective are unchanged. As the target is convex, we can find the minimum by deriving the derivative for each variable $f'_{\tilde{X}}(v)$:

$$\begin{aligned} &\frac{\partial \sum_v (f'_{\tilde{X}}(v) - \tilde{f}_{\tilde{X}}(v))^2 + a + b}{\partial f'_{\tilde{X}}(v)} = 0 \\ \longrightarrow &f'_{\tilde{X}}(v) = \tilde{f}_{\tilde{X}}(v) - \frac{1}{2}(\mu + \lambda_v) \end{aligned} \tag{32}$$

Then, we re-divide the domain D into two sub-domain $D^* \subseteq D$ and $D_* = D \setminus D^*$ such that $\forall v \in D^*, f'_{\tilde{X}}(v) = 0$ and $\forall v \in D_*, f'_{\tilde{X}}(v) > 0 \wedge \lambda_v = 0$. For all $v \in D$, we have

$$1 = \sum_{v \in D^*} f'_{\tilde{X}}(v) + \sum_{v \in D_*} f'_{\tilde{X}}(v) = \sum_{v \in D_*} \tilde{f}_{\tilde{X}}(v) - \frac{|D_*|\mu}{2} \tag{33}$$

From this, we can derive μ as follows.

$$\mu = \frac{2}{|D_*|}(\sum\nolimits_{v \in D_*} \tilde{f}_{\tilde{X}}(v) - 1) \tag{34}$$

By plugging μ into Equation (32), for all $v \in D_*$ we have $f'_{\tilde{X}}(v)$ as follows:

$$f'_{\tilde{X}}(v) = \tilde{f}_{\tilde{X}}(v) - \frac{1}{|D_*|}(\sum\nolimits_{v \in D_*} \tilde{f}_{\tilde{X}}(v) - 1). \tag{35}$$

Implementation of LDPRecover. We provide the pseudo-code for LDPRecover without prior knowledge in Algorithm 1. Specifically, we first estimate the genuine frequencies (lines 1-4): we initialize D_0, D_1, compute the malicious frequencies $\tilde{f}'_Y(v)$ for each $v \in D$ and estimate the genuine frequencies $f'_{\tilde{X}}(v)$ for each $v \in D$. Then, we through an iterative process of finding D^* to refine the recovered frequencies based on the genuine frequencies (lines 5-11): we initiate $D^* = \emptyset$ and $D_* = D$, and then iteratively test whether $f'_{\tilde{X}}(v)$ for all $v \in D_*$ are positive. In each iteration, we move v from D_* to D^* if $f'_{\tilde{X}}(v)$ is negative. The iterative procedure is repeated until $f'_{\tilde{X}}(v)$ for all $v \in D_*$ are non-negative. In particular, Algorithm 1 becomes the pseudo-code for LDPRecover with partial knowledge by replacing $\tilde{f}'_Y(v)$ and Equations (26) and (27) with $\tilde{f}^*_Y(v)$ and Equations (30) and (31).

E. Approximation Error of LDPRecover

LDPRecover is built on the genuine frequency estimator, which assumes the server receives sufficient reports from genuine and malicious users. When this assumption no longer holds, the *Lindeberg–Lévy* CLT theorem provides an asymptotic approximation of the genuine frequency estimator. To understand the gap between the asymptotic estimator and the true one, we analyze the approximation error of $\tilde{f}_{\tilde{X}}(v)$ and $\tilde{f}_Y(v)$ in our estimator in terms of the numbers of genuine and fake users, i.e., n and m.

Specifically, for $\tilde{f}_Y(v)$, suppose its true *probability density function* (pdf) is $\tilde{\theta}_{Y,v}$, then its *cumulative distribution function* (cdf) would be $\tilde{\Theta}_{Y,v}(w) = \int_{-\infty}^{w} \tilde{\theta}_{Y,v}(\tilde{f}_Y(v))d(\tilde{f}_Y(v))$. According to Lemma 1, the approximated *pdf* of $\tilde{f}_Y(v)$ is $\hat{\theta}_{Y,v}(\tilde{f}_Y(v)) = \frac{1}{\sqrt{2\pi}\sigma_y} \exp(-\frac{(\tilde{f}_Y(v)-\mu_y)^2}{2\sigma_y^2})$, and its *cdf* is $\hat{\Theta}_{Y,v}(w) = \int_{-\infty}^{w} \hat{\theta}_{Y,v}(\tilde{f}_Y(v))d(\tilde{f}_Y(v))$. Then we have:

Algorithm 1 LDPRecover

Input: Poisoned frequencies $\tilde{\mathbf{f}}_Z$, estimated sum of malicious frequencies $\sum_{v \in D} \tilde{f}_Y(v)$, η
Output: Recovered frequencies $\mathbf{f}'_{\tilde{X}}$

1: // Estimate genuine frequencies
2: Initialize $D_0 = \{v | \tilde{f}_Z(v) \leq 0\}$ and $D_1 = D \setminus D_0$
3: Compute $\tilde{f}'_Y(v)$ for all $v \in D$ according to Equation (26)
4: Estimate $\tilde{f}_{\tilde{X}}(v)$ for all $v \in D$ according to Equation (27)
5: // Refine recovered frequencies
6: Initialize $D^* = \emptyset$ and $D_* = D$
7: Initialize $f'_{\tilde{X}}(v)$ for all $v \in D_*$ according to Equation (35)
8: **while** for all $v \in D_*, \min\{f'_{\tilde{X}}(v)\} < 0$ **do**
9: Move $v \in D_*$ from D_* to D^* if $f'_{\tilde{X}}(v) < 0$
10: Update $f'_{\tilde{X}}(v)$ for all $v \in D_*$ according to Equation (35)
11: **end while**
12: **return** $\mathbf{f}'_{\tilde{X}}$

Theorem 4. *For $\tilde{f}_Y(v)$, its true cdf $\tilde{\Theta}_{Y,v}(w)$ and approximated cdf $\hat{\Theta}_{Y,v}(w)$ differ by no more than $\frac{0.33554(g_y + 0.415\sigma_y^3)}{\sigma_y^3\sqrt{m}}$, where $g_y = \mathbf{E}[|\tilde{f}_Y(v) - \mu_y|^3]$, and μ_y, σ_y can be obtained from Lemma 1.*

Proof. Please refer to our technical report [48]. □

Similar to $\tilde{f}_Y(v)$, we have the error bound of $\tilde{f}_{\tilde{X}}(v)$ as follows.

Theorem 5. *For $\tilde{f}_{\tilde{X}}(v)$, its true cdf $\tilde{\Theta}_{\tilde{X},v}(w)$ and approximated cdf $\hat{\Theta}_{\tilde{X},v}(w)$ differ by no more than $\frac{0.33554(g_x + 0.415\sigma_x^3)}{\sigma_x^3\sqrt{n}}$, where $g_x = \mathbf{E}[|\tilde{f}_{\tilde{X}}(v) - \mu_x|^3]$, and μ_x, σ_x can be obtained from Lemma 2.*

In Theorems 4 and 5, the gap between asymptotic gain and true one can be defined as functions of n and m. Therefore, the speed of convergence rate in the asymptotic distributions of $\tilde{f}_Y(v)$ and $\tilde{f}_{\tilde{X}}(v)$ are at least on the order of $\frac{1}{\sqrt{n}}$ and $\frac{1}{\sqrt{m}}$, respectively. That is, the approximation error is still tolerable even if the number of reports is insufficient.

VI. Evaluation

A. Experimental Setup

1) Datasets: We evaluate our proposed LDPRecover on two real-world datasets, i.e., IPUMS [53] and Fire [54].

IPUMS [53]: The IPUMS dataset encompasses U.S. census data collected over several years. For this study, we have chosen to focus on the most recent data from 2017 and have identified the attribute of "city" as the item held by each user. This yields a total of 102 items across 389,894 users.

Fire [54]: The Fire dataset, collated by the San Francisco Fire Department on January 16, 2023, documents details pertaining to service calls. We refine these records based on call type and utilize data corresponding to the "Alarms" category. The "unit ID" is considered as the item held by each user, culminating in a total of 490 items and 667,574 users.

2) LDP Settings: We consider three popular LDP protocols for frequency estimation, namely GRR, OUE, and OLH, the details of which are given in Section III-B. We set the default parameters of the LDP protocols as $\epsilon = 0.5$ and $g = \lceil e^{\epsilon} + 1 \rceil$.

3) Attack Settings: We consider two state-of-the-art poisoning attacks to LDP, namely **Manip** [8] and **MGA** [9], as well as our proposed **adaptive attack (AA)** already described in Section V-C. For Manip, we first sample a malicious data domain H from the data domain D, and then draw uniform samples (malicious data) from H. For MGA, we randomly select target items and draw samples from the attacker-designed distribution, where the sampling probabilities of all untarget items are 0 and that of the target items are $1/r$ (r is the number of target items). For AA, we randomly generate the attacker-designed distribution, then draw samples from this distribution and use the samples as the malicious users' data.

The parameters involved in the LDP protocols and attacks are $\beta = \frac{m}{n+m}$ (the fraction of malicious users) and r (the number of target items in MGA). We set the default values of these parameters as $\beta = 0.05$ and $r = 10$.

4) Recovery Settings: For ease of presentation, we use LDPRecover and LDPRecover* to denote the versions of LDPRecover operating without and with partial knowledge of the items selected by the attacker, respectively. In LDPRecover, the server aggregates item frequencies without any detail of the attacks. Conversely, LDPRecover* operates under the assumption that the server is aware of the attacker-selected items. In the context of MGA, these items are explicitly identified as target items, while in AA, they are the items that exhibit the top-$r/2$ frequency increase following the attack.

The parameter involved in LDPRecover and LDPRecover* is $\eta = \frac{m}{n}$ (the ratio of the number of malicious users to the number of genuine users). Since, in practice, the server does not know what the real value of η is, we set a large $\eta = 0.2$ by default in the experiments, which is a value larger than the real value (i.e., $\eta \gg \frac{\beta}{1-\beta}$). We also explore the impact of η on recovered results from the poisoning attacks in Section VI-D, where the range of η is $\eta \in [0.01, 0.4]$.

5) Comparison Methods: To the best of our knowledge, LDPRecover is the only method that recovers the accurate aggregated frequencies from the poisoned ones, whether or not the details of the attacks are known. To achieve a fair comparison, we incorporate partial knowledge (in Section V-D) to adapt the detection method. Specifically, Detection identifies users as malicious if their reported data matches the target items.

B. Evaluation Metrics

We adopt mean squared error (MSE) and frequency gain (FG) as evaluation metrics. We define them as follows.

Mean Squared Error (MSE). Given original frequencies and (recovered or poisoned) aggregated frequencies, we use the MSE to evaluate the average error of frequencies for all items. Specifically,

$$\text{MSE} = \frac{1}{d} \sum\nolimits_{v \in D} (\tilde{f}_X(v) - \tilde{f}_Z^*(v))^2, \tag{36}$$

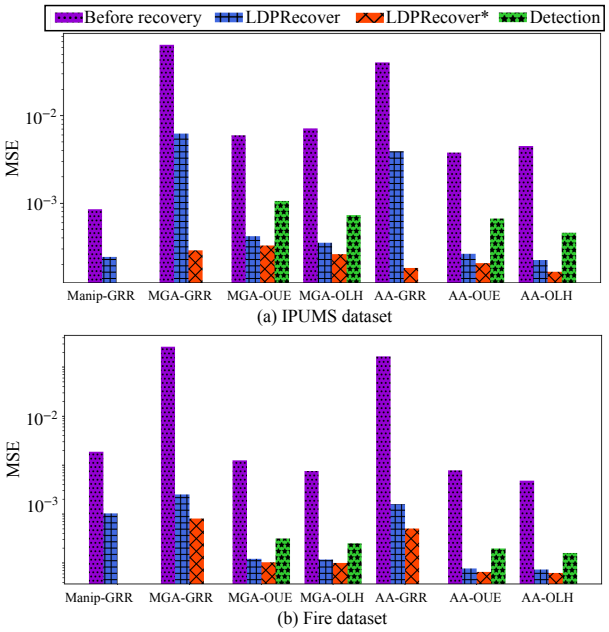

Fig. 3. The mean squared error (MSE) of LDPRecover and LDPRecover* for two datasets, three LDP protocols, and three attacks. "-Manip", "-MGA", and "-AA" represents the results for recovery from Manip, MGA, and AA, respectively.

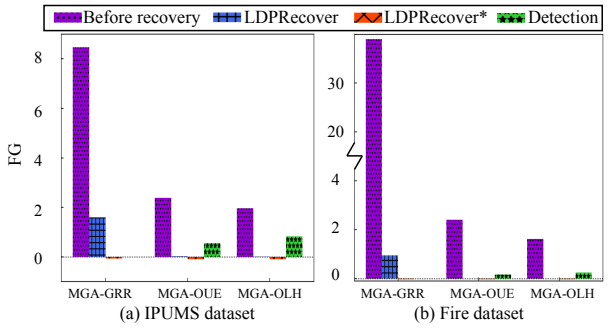

Fig. 4. The frequency gain (FG) of LDPRecover and LDPRecover* for two datasets, three LDP protocols, and three attacks. "-Manip", "-MGA", and "-AA" represents the results for recovery from Manip, MGA, and AA, respectively.

where d is the size of the domain D, $\tilde{f}_X(v)$ is the original frequency of v in genuine data X, and $\tilde{f}_Z^*(v)$ could be the recovered or poisoned frequency of v.

Frequency Gain (FG). For targeted poisoning attacks, we follow [9] to evaluate the recovery methods via FG. Specifically, given a set of target items $\mathcal{T}$, FG is defined as the sum of frequency gain of each target item $t \in \mathcal{T}$, i.e.,

$$\text{FG} = \sum\nolimits_{t \in \mathcal{T}} (\tilde{f}_{\tilde{X}}(t) - \tilde{f}_Z^*(t)), \tag{37}$$

where $\tilde{f}_{\tilde{X}}(v)$ is the genuine frequency of v aggregated from genuine data X using LDP protocols.

We say a recovery method is more accurate and effective if the recovered frequencies have a smaller MSE and FG: smaller MSE means better accuracy, and smaller FG implies better counter targeted attacks. Since the frequency recovery process

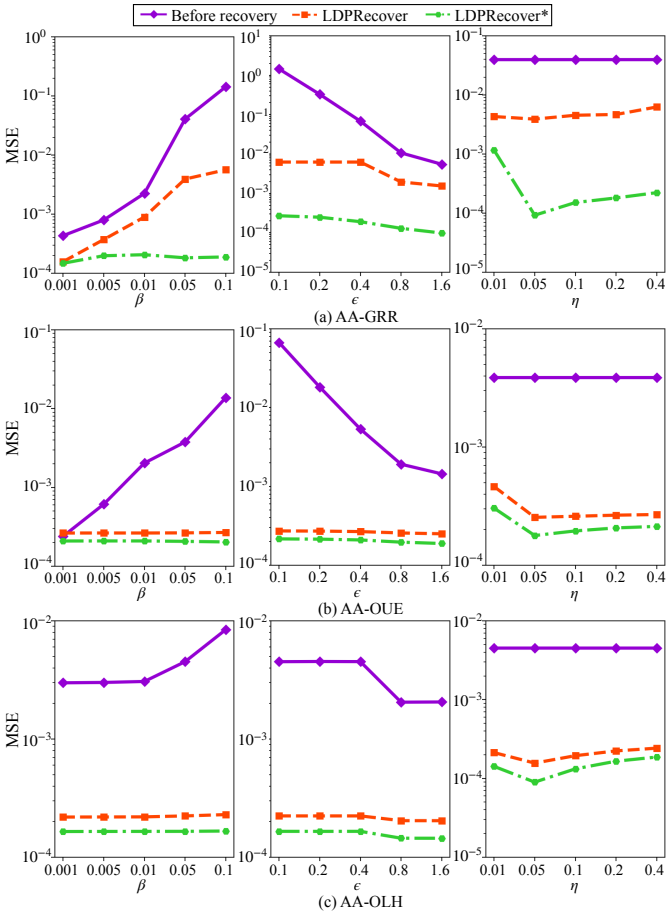

Fig. 5. Impact of differential parameters (β, ϵ, η) on recovery from AA on IPUMS dataset in terms of MSE.

involves randomness, we average the results over 10 trials to compute the MSE and FG in the experiments.

C. Experimental Results

Figures 3 and 4 show the MSE and FG of LDPRecover and LDPRecover* for the two datasets, three LDP protocols, and three attacks. From these figures, we have the following observations:

Both LDPRecover and LDPRecover* are Accurate and Widely Applicable. Both LDPRecover and LDPRecover* can recover the accurate aggregated frequencies from the poisoned ones caused by Manip, MGA, and AA, even if the default $\eta = 0.2$ significantly exceeds the actual ratio $\frac{\beta}{1-\beta} = 0.052$. Notably, LDPRecover* consistently performs best when countering MGA, i.e., it achieves a lower MSE than LDPRecover. Specifically, both LDPRecover and LDPRecover* recover aggregated frequencies by subtracting the malicious frequencies from the poisoned frequencies, i.e., the more accurate the estimated malicious frequencies, the more accurate the recovered frequencies are. Based on this, while LDPRecover estimates these malicious frequencies based solely on the LDP protocol's properties, LDPRecover* refines these estimations by incorporating information about the attacker-selected items, thus enhancing accuracy. To confirm this, we evaluated the

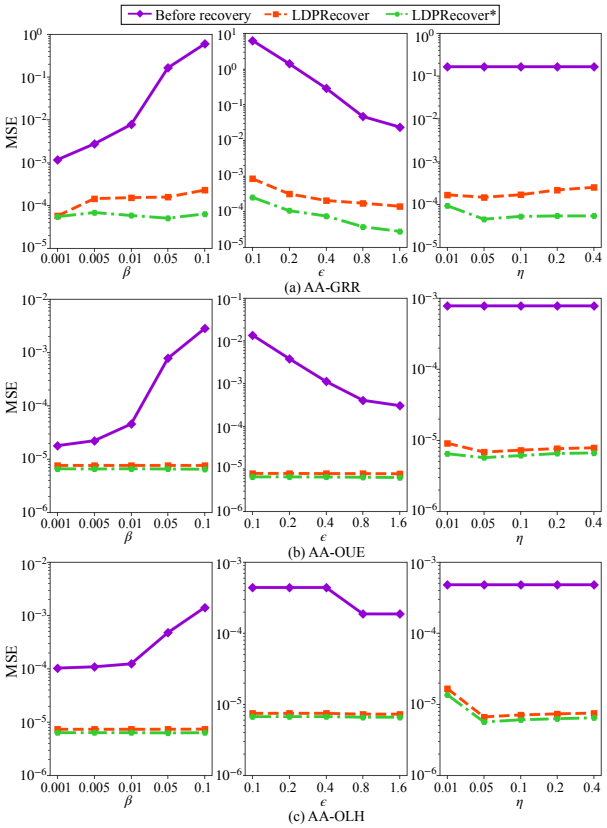

Fig. 6. Impact of differential parameters (β, ϵ, η) on recovery from AA on fire dataset in terms of MSE.

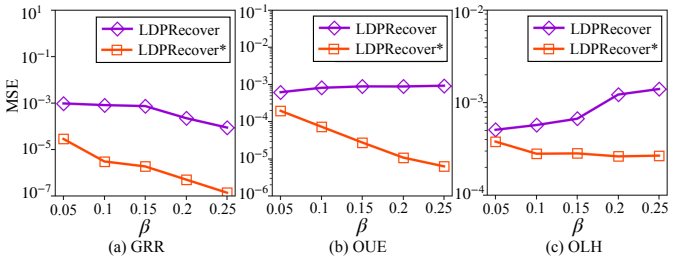

Fig. 7. MSE between LDPRecover and LDPRecover* estimated malicious frequencies and true malicious frequencies (IPUMS).

MSEs of the malicious frequencies estimated by LDPRecover and LDPRecover* versus the true malicious frequencies. The experimental results in Figure 7 show that LDPRecover* estimates malicious frequencies more accurately than LDPRecover. Consequently, LDPRecover* surpasses LDPRecover in recovering the aggregated frequencies by estimating more accurate malicious frequencies. In other words, the prior knowledge of attacker-selected items introduced in LDPRecover* helps to improve the accuracy of the recovered frequency. In addition, we note that LDPRecover and LDPRecover* outperform Detection in all cases. This is because Detection removes all users with the target items indiscriminately, such that the genuine users with the target items are incorrectly removed.

Both LDPRecover and LDPRecover* reduce FG of Tar-

TABLE I
MSE OF THE LDPRECOVER EXECUTION ON THE UNPOISONED FREQUENCIES

LDP	IPUMS		Fire	
	Before-Rec	After-Rec	Before-Rec	After-Rec
GRR	5.89E-4	5.31E-4	1.68E-3	3.62E-5
OUE	3.81E-5	5.33E-4	2.93E-5	3.64E-5
OLH	1.21E-6	5.30E-4	6.87E-7	3.63E-5

geted Attacks. LDPRecover and LDPRecover* can effectively defend against targeted attacks. First, LDPRecover can significantly reduce the FG of targeted attacks, especially to almost 0 in most cases. Second, LDPRecover* even reduces the FG to a negative value (FG<0), which means that the recovered frequencies of the target items are even small than the ones before the attack. Third, Detection is much less effective in dealing with targeted attacks than LDPRecover, not to mention LDPRecover*. This performance gap comes from the same reason: Detection brutally removes all users with target items. In contrast, LDPRecover*, which has the same prior knowledge as Detection, can obtain more accurate malicious data, as shown in Figure 7. This allows LDPRecover* to finely deduct malicious data from poisoned data, gaining a lower FG.

D. Effectiveness of Various Parameters

In this subsection, we aim to study the impact of different parameter settings (i.e., β, ϵ, and η) on the recovery results of our recovery methods from adaptive attacks. In particular, β, ϵ are the parameters of the poisoning attacks, while η is the parameter of our recovery methods. Specifically, we vary one parameter while keeping the others fixed to their default values to investigate its impact on our recovery methods, where the range of these parameters are $\beta = 0$ (i.e., LDPRecover executes on the unpoisoned data), $\beta \in [0.001, 0.1]$, $\epsilon \in [0.1, 1.6]$, and $\eta \in [0.01, 0.4]$.

Impact of the fraction of malicious users β. The first column of Figures 5 and 6 shows the impact of β on recovering frequencies from AA over IPUMS and Fire datasets, respectively. We observe that when defensing against poisoning attacks, LDPRecover and LDPRecover* can recover the accurate aggregated frequencies under various β. Moreover, Table I shows the MSE of the LDPRecover execution on the unpoisoned frequencies, where Before-Rec and After-Rec denote the MSE of unpoisoned frequencies and the MSE of the frequencies recovered by LDPRecover on the unpoisoned frequencies, respectively. From this table, we observe that when LDPRecover executes on the unpoisoned frequencies, it can improve the accuracy of the frequencies from GRR and reduce the accuracy of the frequencies from OLH and OLH. This is because when LDPRecover executes on the unpoisoned frequencies from OUE and OLH, it may remove frequencies that should not be removed, thus reducing accuracy.

Impact of the privacy budget of the LDP protocols ϵ. The second column of Figures 5 and 6 shows the impact of the privacy budget on recovering frequencies from AA over IPUMS and Fire datasets, respectively. We observe that regardless of ϵ, LDPRecover and LDPRecover* are both effective in recovering accurate aggregated frequencies. In particular, for LDPRecover*, the MSE remains low and stable under in all cases; for LDPRecover, the MSE may decreases or remains stable as ϵ grows. The discrepancy arises because LDPRecover* exploits known details of the poisoning attack, resulting in a stable and accurate estimation. In contrast, LDPRecover utilizes the LDP protocol's properties for estimation, rendering it susceptible to fluctuations caused by ϵ.

Impact of the ratio of the number of malicious users to the number of genuine users η. The third column of Figures 5 and 6 shows the effect of η on LDPRecover and LDPRecover*. We observe that LDPRecover and LDPRecover* perform optimally when η closely matches β. This is because a more accurate η will enable our methods to estimate genuine frequencies more accurately. Furthermore, they still maintain effectiveness even there is a deviation between η and β. This is because, even if the estimated genuine frequencies are moderately accurate, our method can still refine accurate recovered frequencies from the estimated genuine frequencies using public available knowledge (i.e., Equations (22) and (23)). For example, Figure 5 (a) illustrates that with $\beta = 0.05$ and $\eta = 0.4$, LDPRecover significantly outperforms the poisoned data, with the average MSE of 1.42×10^{-4} for LDPRecover versus 8.78×10^{-2} for the poisoned frequencies.

Overall, these observations illustrate the effectiveness and applicability of our proposed recovery methods.

VII. DISCUSSION

A. Applicability to Other Aggregation Functions

LDPRecover is designed for frequency estimation in LDP protocols. Its effectiveness is based on the principle that frequencies aggregated by LDP protocols are normally distributed when the number of users is sufficiently large, as supported by Theorem 1. This foundational principle ensures that if other aggregation functions (e.g., mean estimation) can be decomposed to frequency estimation problems, LDPRecover retains its effectiveness. As a case in point, consider Harmony [55], a common LDP protocol for mean estimation. Harmony discretizes numerical values into binary categories (e.g., 1 or -1) and applies Randomized Response (an LDP protocol for frequency) for perturbation, subsequently aggregating these perturbed frequencies to compute the mean. Since Harmony follows the frequency estimation paradigm, LDPRecover is effective for Harmony.

B. Extension to Defend against Input Poisoning Attacks

Note that our threat model essentially follows the general poisoning attack [8], [9], [56], [57], where malicious users can send attacker-crafted data directly to the server, bypassing LDP perturbation mechanisms. Some works [8], [9], [56], [57] have also explored the input poisoning attack (IPA) you mentioned, where malicious users strictly follow the LDP perturbation. While IPA makes frequency recovery more challenging due to perturbation, it is far less effective than general poisoning attacks, as highlighted by the majority

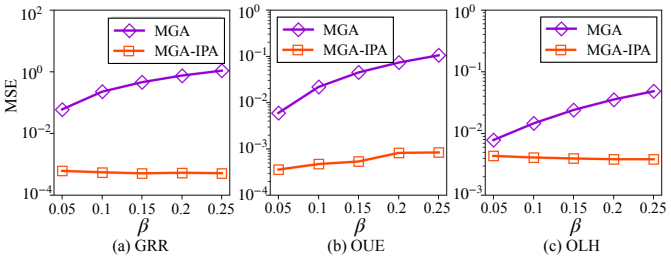

Fig. 8. Comparison of MGA performance under the general poisoning attack and IPA (IPUMS).

of existing studies [8], [9], [56], [57]. To verify this, we implemented MGA under IPA (denoted as MGA-IPA) by sending the MGA-produced malicious data to the server after perturbing it via LDP perturbation mechanisms, and then compared its performance difference with the original MGA (under the general poisoning attack). The experimental results in Figure 8 show that MGA-IPA performs significantly worse than the original MGA (under the general poisoning attack). For example, when attacking GRR, the MSE of original MGA is between $6.07 \times 10^{-2} \sim 1.08$ while that of MGA-IPA alone is between $5.16 \times 10^{-4} \sim 6.21 \times 10^{-4}$, resulting in an improvement of $2 \sim 4$ orders of magnitude.

Although our primary focus is on defending against general poisoning attacks, LDPRecover can be adapted to counteract IPA by integrating existing detection methods, such as k-means clustering approach [56], [57]. The k-means-based defense samples multiple subsets from users and clusters these subsets into two clusters: the cluster with more subsets is considered as the genuine cluster and are used to frequency estimation, while the other cluster is considered as the malicious cluster. Note that under IPA, we cannot estimate the statistics of malicious frequencies in the same way as in this work (e.g., Equation (21)), since the statistics of malicious data align with the genuine data. To address this problem, we integrate k-means-based defense into LDPRecover (denoted by LDPRecover-KM): we use k-means-based defense to estimate the cluster of malicious users and learn statistics of malicious frequencies from the cluster, making the proposed LDPRecover still effective. The results are shown in Figure 9, where ξ is the sample rate of k-means-based defense. This figure shows that by such integration, LDPRecover can recover accurate aggregated frequencies under IPA. For example in Figure 9 (a), when MGA-IPA attacks GRR, the integration of LDPRecover with k-means clustering yields a 48.9% improvement in recovery accuracy compared to using k-means clustering alone.

C. Applicability to Multi-Attacker Case

As a final note, LDPRecover can also be used to defend against the multi-attacker threat model, in which multiple attackers control different groups of malicious users. Specifically, under the adaptive attack in LDPRecover, multiple attackers sampling malicious data from their respective attacker-designed distributions can be viewed as one attacker sampling malicious data from the joint distribution of these distributions. That is, the multi-attacker threat model can be treated as

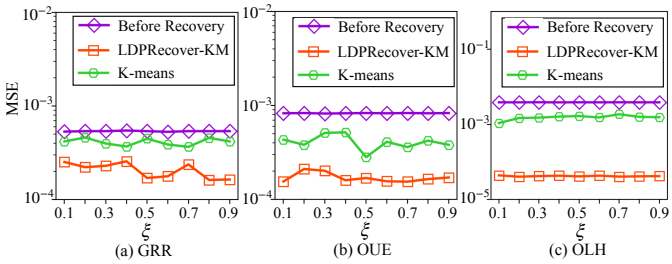

Fig. 9. Comparison of LDPRecover-KM and k-means performance under MGA-IPA (IPUMS).

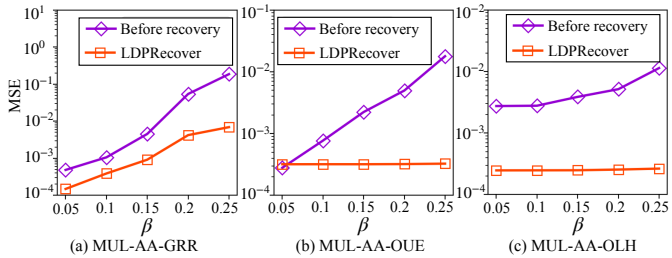

Fig. 10. The performance of LDPRecover when dealing with multi-attacker poisoning attacks (IPUMS).

a case of the original threat model (in Section IV.A), so LDPRecover is still effective for this case. To verify this, we conduct the following experiments to evaluate the performance of LDPRecover when dealing with multiple attackers. In the experiment, we set up five attackers to perform AA and randomly assign malicious users to these attackers. Figure 10 demonstrates that LDPRecover accurately recovers aggregated frequencies from multi-attacker poisoned data. For example in Figure 10 (a), LDPRecover achieves an average improvement of 80.2% in the accuracy of aggregated frequencies compared to the poisoned data. These results validate our analysis above.

VIII. Conclusion

In this work, we perform the first systematic study on frequency recovery from LDP poisoning attacks. Our proposed recovery method, called LDPRecover, can eliminate the impact of poisoning attacks on the aggregated frequencies gathered from LDP protocols. In particular, LDPRecover can be used as a frequency recovery paradigm that can enhance the overall accuracy of the recovered frequencies by integrating the attack details into LDPRecover. Our experimental results confirm the effectiveness of the proposed method at recovering aggregated frequencies from the poisoned ones.

An interesting future work is to extend LDPRecover to poisoning attacks on LDP protocols for more complex tasks, such as key-value pairs collection under LDP.

Acknowledgment

This work was supported by the National Key Research and Development Program of China (No. 2022ZD0120200), the National Natural Science Foundation of China (Grant No: 62102334, 92270123 and 62372122), and the Research Grants Council, Hong Kong SAR, China (Grant No: 15225921, 15209922, 15208923, 15210023 and C2004-21GF), the Fundamental Research Funds for the Central Universities (NO. 501QYJC2023121001).

REFERENCES

[1] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *FOCS*, 2013.

[2] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *TCC*, 2006.

[3] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, 2014.

[4] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *CCS*, 2014.

[5] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries," 2016.

[6] A. Bittau, Ú. Erlingsson, P. Maniatis, I. Mironov, A. Raghunathan, D. Lie, M. Rudominer, U. Kode, J. Tinnes, and B. Seefeld, "Prochlo: Strong privacy for analytics in the crowd," in *SOSP*, 2017.

[7] A. D. P. Team, "Learning with privacy at scale," *Machine Learning Journal*, 2017.

[8] A. Cheu, A. Smith, and J. Ullman, "Manipulation attacks in local differential privacy," in *S&P*, 2021.

[9] X. Cao, J. Jia, and N. Z. Gong, "Data poisoning attacks to local differential privacy protocols," in *USENIX Security*, 2021.

[10] Y. Wu, X. Cao, J. Jia, and N. Z. Gong, "Poisoning attacks to local differential privacy protocols for key-value data," 2022.

[11] T. Kieu, B. Yang, and C. S. Jensen, "Outlier detection for multidimensional time series using deep neural networks," in *MDM*, 2018.

[12] Y. Zhou, H. Zou, R. Arghandeh, W. Gu, and C. J. Spanos, "Non-parametric outliers detection in multiple time series a case study: Power grid data analysis," in *AAAI*, 2018.

[13] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *KDD*, 2019.

[14] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," in *NeurIPS*, 2014.

[15] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *USENIX Security*, 2017.

[16] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, 1965.

[17] R. Bassily and A. Smith, "Local, private, efficient protocols for succinct histograms," in *STOC*, 2015.

[18] P. Kairouz, K. Bonawitz, and D. Ramage, "Discrete distribution estimation under local privacy," in *International Conference on Machine Learning*. PMLR, 2016, pp. 2436–2444.

[19] Z. Zhang, T. Wang, N. Li, S. He, and J. Chen, "Calm: Consistent adaptive local marginal for marginal release under local differential privacy," in *CCS*, 2018.

[20] J. Jia and N. Z. Gong, "Calibrate: Frequency estimation and heavy hitter identification with local differential privacy via incorporating prior knowledge," in *INFOCOM*, 2019.

[21] T. Wang, M. Lopuhaä-Zwakenberg, Z. Li, B. Skoric, and N. Li, "Locally differentially private frequency estimation with consistency," in *NDSS*, 2020.

[22] Q. Xue, Q. Ye, H. Hu, Y. Zhu, and J. Wang, "Ddrm: A continual frequency estimation mechanism with local differential privacy," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[23] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *NeurIPS*, 2017.

[24] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Minimax optimal procedures for locally private estimation," *Journal of the American Statistical Association*, 2018.

[25] N. Wang, X. Xiao, Y. Yang, J. Zhao, S. C. Hui, H. Shin, J. Shin, and G. Yu, "Collecting and analyzing multidimensional data with local differential privacy," in *ICDE*, 2019.

[26] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, N. Li, and B. Škoric, "Estimating numerical distributions under local differential privacy," in *SIGMOD*, 2020.

[27] J. Duan, Q. Ye, and H. Hu, "Utility analysis and enhancement of ldp mechanisms in high-dimensional space," 2022.

[28] R. Bassily, K. Nissim, U. Stemmer, and A. G. Thakurta, "Practical locally private heavy hitters," in *NeurIPS*, 2017.

[29] T. Wang, N. Li, and S. Jha, "Locally differentially private heavy hitter identification," *TDSC*, 2019.

[30] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *CCS*, 2016.

[31] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *S&P*, 2018.

[32] C. Li, M. Hay, G. Miklau, and Y. Wang, "A data-and workload-aware algorithm for range queries under differential privacy," *PVLDB*, 2014.

[33] G. Cormode, T. Kulkarni, and D. Srivastava, "Answering range queries under local differential privacy," *PVLDB*, 2019.

[34] J. Yang, T. Wang, N. Li, X. Cheng, and S. Su, "Answering multi-dimensional range queries under local differential privacy," *PVLDB*, 2020.

[35] X. He, G. Cormode, A. Machanavajjhala, C. M. Procopiuc, and D. Srivastava, "Dpt: differentially private trajectory synthesis using hierarchical reference systems," *PVLDB*, 2015.

[36] R. Chen, H. Li, A. K. Qin, S. P. Kasiviswanathan, and H. Jin, "Private spatial data aggregation in the local setting," in *ICDE*, 2016.

[37] Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren, "Generating synthetic decentralized social graphs with local differential privacy," in *CCS*, 2017.

[38] B. Avent, A. Korolova, D. Zeber, T. Hovden, and B. Livshits, "BLENDER: Enabling local search with a hybrid differential privacy model," in *USENIX Security*, 2017.

[39] G. Cormode, T. Kulkarni, and D. Srivastava, "Marginal release under local differential privacy," in *SIGMOD*, 2018.

[40] X. Ren, C.-M. Yu, W. Yu, S. Yang, X. Yang, J. A. McCann, and S. Y. Philip, "LoPub: High-dimensional crowdsourced data publication with local differential privacy," *TIFS*, 2018.

[41] M. E. Gursoy, L. Liu, S. Truex, L. Yu, and W. Wei, "Utility-aware synthesis of differentially private and attack-resilient location traces," in *CCS*, 2018.

[42] T. Wang, B. Ding, J. Zhou, C. Hong, Z. Huang, N. Li, and S. Jha, "Answering multi-dimensional analytical queries under local differential privacy," in *SIGMOD*, 2019.

[43] Q. Ye, H. Hu, M. H. Au, X. Meng, and X. Xiao, "Towards locally differentially private generic graph metric estimation," in *CDE*, 2020.

[44] Q. Ye, H. Hu, M. H. Au, X. Meng, and X. Xiao, "Lf-gdpr: A framework for estimating graph metrics with local differential privacy," *TKDE*, 2020.

[45] T. Cunningham, G. Cormode, H. Ferhatosmanoglu, and D. Srivastava, "Real-world trajectory sharing with local differential privacy," *PVLDB*, 2021.

[46] J. Shanthikumar and U. Sumita, "A central limit theorem for random sums of random variables," *Operations Research Letters*, 1984.

[47] H. Fischer, *A history of the central limit theorem: from classical to modern probability theory*. Springer.

[48] X. Sun, Q. Ye, H. Hu, J. Duan, T. Wo, J. Xu, and R. Yang, "Ldprecover: Recovering frequencies from poisoning attacks against local differential privacy," Tech. Rep. [Online]. Available: https://www.dropbox.com/scl/fo/88to91xn1u368u6dxoyje/h?rlkey=ttp8zopq0htbkrcbdyeuieqwm&dl=0

[49] R. Chen, H. Li, S. Kasiviswanathan, and H. Jin, "Private dataaggregation framework for untrusted servers," Mar. 23 2021, uS Patent 10,956,603.

[50] M. Hay, V. Rastogi, G. Miklau, and D. Suciu, "Boosting the accuracy of differentially-private histograms through consistency," *PVLDB*, 2010.

[51] W. Karush, "Minima of functions of several variables with inequalities as side constraints," *M. Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago*, 1939.

[52] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Traces and emergence of nonlinear programming*, 2014.

[53] S. Ruggles, S. Flood, R. Goeken, M. Schouweiler, and M. Sobek, "Ipums usa: Version 12.0 [dataset]. minneapolis, mn: Ipums, 2022," *https://doi.org/10.18128/D010.V12.0*, 2022.

[54] "San francisco fire department calls for service," *http://bit.ly/336sddL*, 2023.

[55] T. T. Nguyên, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin, "Collecting and analyzing data from smart device users with local differential privacy," *arXiv preprint arXiv:1606.05053*, 2016.

[56] X. Li, N. Li, W. Sun, N. Z. Gong, and H. Li, "Fine-grained poisoning attack to local differential privacy protocols for mean and variance estimation," in *USENIX Security'23*, 2023, pp. 1739–1756.

[57] R. Du, Q. Ye, Y. Fu, H. Hu, J. Li, C. Fang, and J. Shi, "Differential aggregation against general colluding attackers," in *ICDE*, 2023.