Fine-grained Poisoning Attack to Local Differential Privacy Protocols for Mean and Variance Estimation

Xiaoguang Li[1,2*] Ninghui Li[2] Wenhai Sun[2] Neil Zhenqiang Gong[3] Hui Li[1]

[1]*Xidian University,* [2]*Purdue University,* [3]*Duke University*

Abstract

Although local differential privacy (LDP) protects individual users' data from inference by an untrusted data curator, recent studies show that an attacker can launch a data poisoning attack from the user side to inject carefully-crafted bogus data into the LDP protocols in order to maximally skew the final estimate by the data curator.

In this work, we further advance this knowledge by proposing a new *fine-grained* attack, which allows the attacker to fine-tune and simultaneously manipulate mean and variance estimations that are popular analytical tasks for many real-world applications. To accomplish this goal, the attack leverages the characteristics of LDP to inject fake data into the output domain of the local LDP instance. We call our attack the output poisoning attack (OPA). We observe a security-privacy consistency where a small privacy loss enhances the security of LDP, which contradicts the known security-privacy trade-off from prior work. We further study the consistency and reveal a more holistic view of the threat landscape of data poisoning attacks on LDP. We comprehensively evaluate our attack against a baseline attack that intuitively provides false input to LDP. The experimental results show that OPA outperforms the baseline on three real-world datasets. We also propose a novel defense method that can recover the result accuracy from polluted data collection and offer insight into the secure LDP design.

1 Introduction

Local differential privacy (LDP) [7], a variant of differential privacy [10] in a distributed environment, protects individual user data against an untrusted data collector regardless of the adversary's background knowledge. Numerous LDP protocols have been proposed for various statistical tasks such as frequency [11, 42, 43, 44, 45], mean/variance [8, 41] and distribution [23, 27]. LDP has also been integrated into many real-world applications as a *de facto* privacy-preserving data collection tool. For example, Google deployed LDP in Chrome browser to collect users' homepages [11]; Microsoft implemented LDP in Windows 10 to analyze application usage statistics of customers [6].

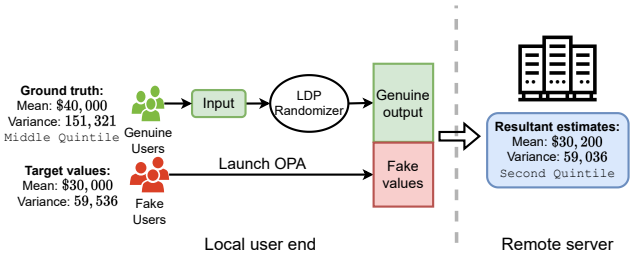


Figure 1: Illustration of our fine-grained data poisoning attacks on LDP-based mean/variance estimation.

Recently, Cao *et al.* [3] and Cheu *et al.* [5] independently studied the security of LDP under *data poisoning attacks* (or called *manipulation attacks* in [5]). They found that LDP randomization is very sensitive to data manipulation such that malicious users could send carefully crafted false data to effectively skew the collector's statistical estimate. In particular, the current attacks aims to "push" the LDP estimate away from the ground truth as far as possible. In [5], the attacker can inject false data through a group of compromised users to degrade the overall LDP performance. The data poisoning attacks in [3, 47] promote the items of interest (e.g., in a recommender system) by maximizing the associated statistical estimates, such as frequency and key-value data.

In this work, we advance the knowledge by considering an attacker who aims to not only manipulate the statistics but also set the estimates to an intended value. We call it a *fine-grained* data poisoning attack. We focus on mean and variance estimation because they are crucial to many data analytical applications in practice. For example, a company conducts a market survey to identify the target market segments based on their customers' income average (the mean) and inequality (the variance) [36] as shown in Figure 1. From the survey, the company estimates the mean and variance of

*This work was done when the author was at Purdue university.

the income so as to make informed decisions on the related services. In order to encourage participation, LDP can be adopted to perturb an individual customer's income value before being sent to the company. Meanwhile, a rival company may control a group of fake responders to launch the *fine-grained* data poisoning attack by submitting erroneous information in hopes of bringing the final estimates as close to their intended values as possible. Consequently, the result deviates from reality and leads to a deceptive conclusion, e.g., the customers in the middle-income quintile are mistakenly believed to come from a lower quintile [12].

We propose an output poisoning attack (OPA) for the *fine-grained* manipulation goal on the local user side against two state-of-the-art LDP protocols for mean and variance, i.e., Stochastic Rounding (SR) [8] and Piecewise Mechanism (PM) [41]. The attack is illustrated in Figure 1. Consistent with prior work, we assume that the attacker can control a group of fake users by purchasing accounts from dark markets [3], and has access to the LDP implementation details. As a result, the attacker can bypass the LDP perturbation and generate bogus values in the output domain of the local LDP instance, which will be sent to the server for final estimation. To demonstrate the effectiveness of OPA, we compare it with a baseline attack – we call it an input poisoning attack (IPA), which represents a straightforward data manipulation by providing fake input to the LDP perturbation without leveraging the LDP protocol.

The main challenge for the attacker here is to manipulate two correlated statistical estimates – mean and variance at the same time through a single LDP query [23]. To address this challenge, we formulate the attack as a simultaneous equation-solving problem and coordinate the generation of the poisonous data across the controlled users. To control the LDP estimate at a finer level, the attack also depends on two observations in reality. First, companies and governments, for commercial, public interest or as required by regulations, need to periodically collect user information to learn the *status quo* and then publish the related statistical results [11, 17, 26, 38]. Second, those historical results regarding the same entity tend to be stable over a short period of time [12, 38, 39]. As a result, the attacker can leverage the data transparency and the predictable information changes to enable fine-grained data manipulation. Specifically, we assume that the attacker can acquire related background information about genuine users from recent public statistical reports or by compromising a small number of users (see Threat model in Section 3).

We systematically study the proposed attack both theoretically and empirically. We first analyze the sufficient conditions to launch the attack and further discuss the lower bound on the required number of fake users given the target mean and variance. The results show that OPA needs fewer fake users than the baseline to achieve the same target values. We are interested in the relationship between various attack parameters and performance, as well as the associated implications. Thus, we also study the MSE between the target value and the final estimate. The results show that OPA has a smaller MSE because direct manipulation of the local LDP output will ignore the effect of perturbation and give the attacker a significant advantage in producing an intended result.

In the literature, a security-privacy trade-off for LDP protocols was revealed: a small ε (strong privacy guarantee) leads to a less secure LDP protocol against prior data poisoning attacks [3, 5, 47]. However, we in this work have an opposite observation that weak privacy protection with a large ε is vulnerable to our fine-grained attack. We call this *security-privacy consistency* for LDP protocols. We analyze the two assertions and show that they surprisingly are both valid and that they together provide a holistic understanding of the threat landscape. This conclusion is disturbing since it complicates the already elusive reasoning and selection of the privacy budget in LDP and makes designing a secure LDP more difficult (see Section 6). To mitigate our attack, we also propose a clustering-based method for bogus data tolerance and discuss the relevant defenses in Section 8. Our main contributions are:

- We are the first to study the *fine-grained data poisoning attack* against the state-of-the-art LDP protocols for mean and variance estimation.
- We propose the *output poisoning attack* to precisely control the statistical estimates to the intended values. By the comparison with an LDP-independent baseline attack, i.e., *input poisoning attack*, we show that OPA can achieve better attack performance by taking advantage of LDP.
- We theoretically analyze the sufficient conditions to launch the proposed attacks, study the attack errors, and discuss the factors that impact the attack effectiveness.
- We discover a fundamental security-privacy consistency in our attacks, which is at odds with the prior finding of the security-privacy trade-off. We provide an in-depth analysis and discussions to reveal the cause of the difference.
- We empirically evaluate our attacks on three real-world datasets. The results show that given the target values, our attacks can effectively manipulate the mean and variance with small errors. We also develop and evaluate a defense method, and provide insights into secure LDP design and other mitigation methods.

2 Background

2.1 Local Differential Privacy

In the local setting of differential privacy, it is assumed that there is no trusted third party. In this paper, we consider there are n users and one remote server. Each user possesses a data value $x \in \mathcal{D}$, and the server wants to estimate the mean and variance of values from all local users. To protect privacy, each user randomly perturbs his/her x using an algorithm

$\Psi(x) : \mathcal{D} \to \widehat{\mathcal{D}}$, where $\widehat{\mathcal{D}}$ is the output domain of Ψ, and sends $x' = \Psi(x)$ to the server.

Definition 1 (ε-Local Differential Privacy (ε-LDP) [7]). *An algorithm $\Psi(\cdot) : \mathcal{D} \to \widehat{\mathcal{D}}$ satisfies ε-LDP ($\varepsilon > 0$) if and only if for any input $x_1, x_2 \in \mathcal{D}$, the following inequality holds:*

$$\forall T \in \widehat{\mathcal{D}}, \quad \Pr[\Psi(x_1) = T] \leq e^{\varepsilon} \Pr[\Psi(x_2) = T].$$

Intuitively, an attacker cannot deduce with high confidence whether the input is x_1 or x_2 given the output of an LDP mechanism. The offered privacy is controlled by ε, i.e., small (large) ε results in a strong (weak) privacy guarantee and a low (high) data utility. Since the user only reports the privatized result $\Psi(x)$ instead of the original value x, even if the server is malicious, the users' privacy is protected. In our attack, *the attacker can manipulate a group of fake users in order to change the estimates of mean/variance on the server* (See Section 3 for the detailed threat model).

2.2 Mean and Variance Estimation with LDP

We introduce two widely-used LDP mechanisms for mean and variance estimation, *Stochastic Rounding (SR)* [8] and *Piecewise Mechanism (PM)* [41]. Note that they were originally developed for mean estimation only and were subsequently adapted to support variance estimation in [23]. In this work, we use the adapted version.

2.2.1 SR mechanism

The SR mechanism first uniformly partitions all users into two groups: group g_1 reports their original values and group g_2 submits their squared original values. All the values must be transformed into $[-1, 1]$ before being used in the LDP.

Perturbation. SR first converts the value into the range $[-1, 1]$. Suppose that the range of the original input values is $[a, b]$. SR calculates transformation coefficients $k_1 = \frac{2}{b-a}$ for g_1 and $k_2 = \frac{2}{|b^2-a^2|}$ for g_2, and derives $\tilde{x} = -1 + k_1(x-a)$ for g_1 or $\tilde{x} = -1 + k_2(x-a^2)$ for g_2. Then SR perturbs values in a discrete output domain with the probability mass function

$$\Pr[\Psi_{SR(\varepsilon)}(\tilde{x}) = x'] = \begin{cases} q + \frac{(p-q)(1-\tilde{x})}{2}, & \text{if } x' = -1 \\ q + \frac{(p-q)(1+\tilde{x})}{2}, & \text{if } x' = 1 \end{cases},$$

where $p = \frac{e^{\varepsilon}}{1+e^{\varepsilon}}$ and $q = 1 - p$.

Aggregation. It has been proven that $\mathbb{E}(\frac{x'}{p-q}) = \tilde{x}$. The server calculates $\Phi_1(x') = (\frac{x'}{p-q} + 1)/k_1 + a$ for g_1 and $\Phi_2(x') = (\frac{x'}{p-q} + 1)/k_2 + a^2$ for g_2, and estimates their mean. The process provides unbiased estimation of the mean of x and x^2, denoted by $\mathbb{E}(x)$ and $\mathbb{E}(x^2)$ respectively. The variance of x is estimated as $\mathbb{E}(x^2) - \mathbb{E}(x)^2$.

2.2.2 PM mechanism

PM also uniformly divides users into groups g_1 and g_2 in which users report the squared values and original values respectively.

Perturbation. In PM, the input domain is $[-1, 1]$ and the output domain is $[-s, s]$, where $s = \frac{e^{\varepsilon/2}+1}{e^{\varepsilon/2}-1}$. Similar to SR, PM first transforms the value into the range $[-1, 1]$ via the same steps in SR. Then PM perturbs each value in the continuous range $[-s, s]$ with the probability density function as follows

$$\Pr[\Psi_{PM(\varepsilon)}(\tilde{x}) = x'] = \begin{cases} \frac{e^{\varepsilon/2}(e^{\varepsilon/2}-1)}{2(e^{\varepsilon/2}+1)}, & \text{if } x' \in [l(\tilde{x}), r(\tilde{x})] \\ \frac{e^{\varepsilon/2}-1}{2(e^{\varepsilon/2}+e^{\varepsilon})}, & \text{otherwise} \end{cases},$$

where $-s \leq l(\tilde{x}) < r(\tilde{x}) \leq s$, $l(\tilde{x}) = \frac{e^{\varepsilon/2}\tilde{x}-1}{e^{\varepsilon/2}-1}$ and $r(\tilde{x}) = \frac{e^{\varepsilon/2}\tilde{x}+1}{e^{\varepsilon/2}-1}$.

Aggregation. It has been proven that $\mathbb{E}(x') = \tilde{x}$ in PM. The server re-converts x' to $\Phi_1(x') = (x'+1)/k_1 + a$ for g_1 and to $\Phi_2(x') = (x'+1)/k_2 + a^2$ for g_2, and then estimates their mean, from which the server can get the unbiased mean estimations $\mathbb{E}(x)$ and $\mathbb{E}(x^2)$. The variance of x is estimated as $\mathbb{E}(x^2) - \mathbb{E}(x)^2$. The following lemma shows the error of the SR and PM mechanisms, which is useful for later analysis of the attack error.

Lemma 1 (Error of SR and PM mechanisms [41]). *Assume there are n users with the values $x_1, ..., x_n$. Let μ be the mean of those values, and $\hat{\mu}_{SR}$ and $\hat{\mu}_{PM}$ be the mean estimated by the SR and PM respectively. The errors of SR and PM are*

$$\mathbb{E}\left[(\hat{\mu}_{SR} - \mu)^2\right] = \frac{1}{n^2(p-q)^2}\left(n - (p-q)^2 \times \sum_{i=1}^{n} x_i^2\right)$$

$$\mathbb{E}\left[(\hat{\mu}_{PM} - \mu)^2\right] = \frac{e^{\varepsilon/2}+3}{3n(e^{\varepsilon/2}-1)^2} + \frac{\sum_{i=1}^{n} x_i^2}{n^2(e^{\varepsilon/2}-1)}.$$

It is also shown in [41] that the PM mechanism has smaller error than the SR mechanism when ε is large.

3 Threat Model

In this section, we present our threat model, including the attacker's capabilities and objectives.

3.1 Assumption

Our attacks rely on the following assumptions. First, we assume that the data collector periodically collects user information to derive the intended statistical results. For privacy concerns, LDP may be adopted. This periodical data collection is important and even mandatory in practice for the update on the *status quo* in order to make informed decisions for relevant activities in the future. For various reasons, such as transparency, research and regulatory compliance [11, 12, 17, 26, 38], the

results will also be made public, thus accessible to the attacker. Second, if the respective data collections are made over a short period of time, the trend of those historical results with respect to the same entity tends to be "stable", i.e., their values are close [12, 38, 39]. Therefore, the attacker can use the statistics from the most recent data report to improve the attack accuracy. Specifically, our attacker needs to estimate the number of authentic users n, the sum of the input values of genuine users $S^{(1)} = \sum_{i=1}^{n} x_i$ and the sum of the squared values of genuine users $S^{(2)} = \sum_{i=1}^{n} x_i^2$. Additionally, we assume that the attacker can inject m fake users into the LDP protocol that already contains n genuine users, thus totaling $n+m$ users in the system. This is consistent with prior work showing that an attacker can inject a large number of fake accounts/users into a variety of web services with minimal cost [3, 47]. Next, we discuss the estimation of the required information.

- **Estimating** n. Denote n_e as the estimate of n. The attacker can deduce n_e from publicly available and reliable sources, e.g., service providers often disclose the number of users under the LDP protection for publicity [11, 26].
- **Estimating** $S^{(1)}$ **and** $S^{(2)}$. Let $S_e^{(1)}$ and $S_e^{(2)}$ be the estimate of $S^{(1)}$ and $S^{(2)}$ respectively. We offer two intuitive estimating methods.

  (1) *From public historical data.* This is the most straightforward way. Given the estimated user number n_e, the historical mean μ_e and variance σ_e^2, the attacker can derive $S_e^{(1)} = n_e \times \mu_e$, $S_e^{(2)} = (\sigma_e^2 + \mu_e^2) \times n_e$.

  (2) *Compromising a small number of genuine users.* The attacker can compromise h out of n genuine users and obtain their original input values $[c_1, ..., c_h]$. This is reasonable in practice for a small number h and also a prerequisite for prior work [5]. Thus the attacker can estimate $S_e^{(1)} = \frac{n_e}{h} \sum_{i=1}^{h} c_i$, $S_e^{(2)} = \frac{n_e}{h} \sum_{i=1}^{h} c_i^2$.

Since LDP protocols deploy perturbation on the local user side, we assume that an OPA attacker can gain access to the LDP implementation details, including related parameters and the output domain of the LDP protocol in order to generate and inject bogus data for manipulation. Given a specific LDP protocol, the OPA attacker also knows the group generation strategy for g_1 and g_2, such that she can 1) determine if the attack would be successful before launching the attack (see in Section 5.1), and 2) craft fake values for each group during the attack (see Section 4.2).

For the baseline attack, we only assume that the attacker knows the input domain of the local LDP instance by taking LDP as a black box protocol.

3.2 Attack Objectives

The attacker's goal is to modify the estimated mean $\hat{\mu}_t$ and variance $\hat{\sigma}_t^2$ through LDP to be as close to the target mean μ_t and variance σ_t^2 as possible. Meanwhile, the attacker wishes to simultaneously manipulate $\hat{\mu}_t$ and $\hat{\sigma}_t^2$. We adopt the adapted versions of PM and SR mechanisms to privately estimate the mean and variance within a single protocol invocation. Note that our attack objective also implicitly covers the situation of maximizing (minimizing) the mean and variance by setting a significantly large (small) target μ_t and σ_t^2. In what follows, we will elaborate on our attacks. Some important notations are summarized in Table 1.

Table 1: Notations.

Notation	Description
n	The number of genuine users
n_e	The attacker-estimated n
m	The number of fake users
β	The fraction of fake users $\frac{m}{m+n}$
g_1	The group reporting the original values $\{x_i\}_{i=1}^{n}$
g_2	The group reporting the squared values $\{x_i^2\}_{i=1}^{n}$
$S_e^{(1)}$	The attacker-estimated $\sum_{i=1}^{n} x_i$
$S_e^{(2)}$	The attacker-estimated $\sum_{i=1}^{n} x_i^2$
μ_t	The attacker's target mean
σ_t^2	The attacker's target variance
k_1	The transformation coefficient for g_1
k_2	The transformation coefficient for g_2

4 Attack Details

4.1 Baseline Attack

We first discuss the baseline attack, *input poisoning attack* (IPA). An IPA attacker does not need to know any details of the underlying LDP protocol. It can submit false data as input to the local LDP instance in order to manipulate the final estimate on the server. Later, we will introduce our attack, *output poisoning attack*, to demonstrate the improved attack performance with knowledge of the LDP protocol compared to the baseline.

Specifically, the goal of IPA is to craft the input values for the controlled fake users in order to alter the mean and variance estimates to be close to the attacker's desired mean μ_t and variance σ_t^2. We generalize the attack for both SR and PM mechanisms. Formally, we denote the original input of n genuine users as $[x_1, ..., x_n]$ $(\forall i : x_i \in [a, b])$, and the crafted input of fake users as $[y_1, ..., y_m]$ $(\forall i : y_i \in [a, b])$. We formulate IPA as finding $[y_1, ..., y_m]$ such that $\frac{\sum_{i=1}^{n} x_i + \sum_{i=1}^{m} y_i}{n+m} = \mu_t$ and $\frac{\sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{m} y_i^2}{n+m} - \mu_t^2 = \sigma_t^2$.

To solve $[y_1, ..., y_m]$, the attacker needs to know $S^{(1)} = \sum_{i=1}^{n} x_i$, $S^{(2)} = \sum_{i=1}^{n} x_i^2$ and n, which can be estimated from published information or by compromising a small number of genuine users as described in Section 3. By substituting $S^{(1)}$, $S^{(2)}$ and n with their estimates $S_e^{(1)}$, $S_e^{(2)}$ and n_e, a set of

desired fake values $[y_1, ..., y_m]$ should satisfy

$$\sum_{i=1}^{m} y_i = (n_e + m)\mu_t - S_e^{(1)} \tag{1}$$

$$\sum_{i=1}^{m} y_i^2 = (n_e + m)(\sigma_t^2 + \mu_t^2) - S_e^{(2)}. \tag{2}$$

We transform Equations (1) and (2) into the following optimization problem and solve it to find valid fake values*.

$$\begin{aligned} \min \quad & \left(\sum_{i=1}^{m} y_i^2 - (n_e + m)(\sigma_t^2 + \mu_t^2) - S_e^{(2)} \right)^2 \\ \text{s.t.} \quad & \sum_{i=1}^{m} y_i = (n_e + m)\mu_t - S_e^{(1)}, \forall i : -1 \le y_i \le 1 \end{aligned} \tag{3}$$

4.2 Output Poisoning Attack

We introduce our output poisoning attack that leverages the LDP implementation details to craft the output of the local perturbation in order to set the final estimates to the target mean μ_t and variance σ_t^2.

Let the number of genuine users in g_1 and g_2 be n_1 and n_2, and the number of fake users be m_1 and m_2 respectively. Denote the input of the genuine users in g_i as $x_{1,(i)}, ..., x_{n_i,(i)}$ and the input of the fake users in g_i as $y_{1,(i)}, ..., y_{m_i,(i)}$. Because of the randomness in the LDP local output, the objective of OPA is to produce fake values $\Psi(y_i)$ $\forall i : 1, ..., m$ such that the expected mean and variance are the attacker-intended μ_t and σ_t^2 respectively. However, it is difficult to calculate $\mathbb{E}[\hat{\mu}_t^2]$ because $\mathbb{E}[\hat{\mu}_t^2] = Var[\hat{\mu}_t] + \mathbb{E}[\hat{\mu}]^2$ and $Var[\hat{\mu}_t]$ depends on true data. To address this problem, we slack the attack goal by replacing $\mathbb{E}[\hat{\mu}_t^2]$ with μ_t^2. Formally, we intend to achieve the following attack objective in practice.

$$\mathbb{E}\left[\frac{2}{n+m} \left(\sum_{i=1}^{n_1} \Phi_1(\Psi(x_{i,(1)})) + \sum_{i=1}^{m_1} \Phi_1(\Psi(y_{i,(1)})) \right) \right] = \mu_t$$

$$\Rightarrow \frac{2}{m+n} \left[\frac{1}{2} S^{(1)} + \sum_{i=1}^{m_1} \Phi_1(\Psi(y_{i,(1)})) \right] = \mu_t \tag{4}$$

$$\mathbb{E}\left[\frac{2}{n+m} \left(\sum_{i=1}^{n_2} \Phi_2(\Psi(x_{i,(2)}^2)) + \sum_{i=1}^{m_2} \Phi_2(\Psi(y_{i,(2)}^2)) \right) \right] - \mathbb{E}[\hat{\mu}_t^2] = \sigma_t^2$$

$$\Rightarrow \frac{2}{m+n} \left[\frac{1}{2} S^{(2)} + \sum_{i=1}^{m_2} \Phi_2(\Psi(y_{i,(2)}^2)) \right] - \mu_t^2 = \sigma_t^2 \tag{5}$$

Since the perturbation $\Psi()$ and aggregation $\Phi()$ are different for SR and PM, the remainder of this subsection will study how to solve Equations (4) and (5) and generate the fake values accordingly.

*In this work, we use the Adam optimizer in PyTorch framework [33] to solve the problem (3) with the learning rate 0.001 and 10,000 iterations.

4.2.1 OPA against SR

By substituting n, $S^{(1)}$ and $S^{(2)}$ in Equations (4) and (5) with their estimates n_e, $S_e^{(1)}$ and $S_e^{(2)}$, we have

$$\sum_{i=1}^{m_1} \Psi(y_{i,(1)}) = (p-q) \left[k_1 \left(\frac{n_e + m}{2} \mu_t - \frac{S_e^{(1)}}{2} - m_1 a \right) - m_1 \right] \tag{6}$$

$$\sum_{i=1}^{m_2} \Psi(y_{i,(2)}^2) = (p-q) \times \left[k_2 \left(\frac{n_e + m}{2} (\sigma_t^2 + \mu_t^2) - \frac{S_e^{(2)}}{2} - m_2 a^2 \right) - m_2 \right] \tag{7}$$

where k_1 and k_2 are the transformation coefficients and a is the lower bound of the input range. In SR, the output is either -1 or 1. Consequently, the attacker can prepare the fake values by determining how many "-1" and "1" respectively to be assigned to the fake users. Suppose in group g_i there are $[-1]_{g_i}$ fake users with -1 and $[1]_{g_i}$ fake users with 1. Per Equations (6) and (7), we have

$$\begin{cases} [1]_{g_1} + [-1]_{g_1} = m_1 \\ [1]_{g_1} - [-1]_{g_1} = \sum_{i=1}^{m_1} \Psi(y_{i,(1)}) \end{cases} \begin{cases} [1]_{g_2} + [-1]_{g_2} = m_2 \\ [1]_{g_2} - [-1]_{g_2} = \sum_{i=1}^{m_2} \Psi(y_{i,(2)}). \end{cases}$$

Since there are two unknown variables and two equations, the attacker can solve the above equations to derive the number of 1 and -1 in each group and then randomly assigns them to the fake users in g_1 and g_2.

4.2.2 OPA against PM

In PM, the output value is in the range $[-s, s]$. According to Equations (4) and (5), the attacker can calculate the fake values by solving the following equations

$$\begin{aligned} & \sum_{i=1}^{m_1} \Psi(y_{i,(1)}) = k_1 \left(\frac{n+m}{2} \mu_t - \frac{S_e^{(1)}}{2} - m_1 a \right) - m_1 \\ & \sum_{i=1}^{m_2} \Psi(y_{i,(2)}^2) = k_2 \left(\frac{n+m}{2} (\sigma_t^2 + \mu_t) - \frac{S_e^{(2)}}{2} - m_2 a^2 \right) - m_2 \\ & \forall i : \Psi(y_{i,(1)}), \Psi(y_{i,(2)}^2) \in [-s, s]. \end{aligned}$$

An intuitive method to solve the above equations is to divide the right-hand-side by m_1 or m_2. However, because the fake values generated by this method are equal, the server can easily detect the fake users because it is statistically unlikely that many genuine users will send the same perturbed values. To address this problem, the attacker first solves the equations using the method described above, and then randomly perturbs each value while maintaining the sum and keeping the values in $[-s, s]$. Finally, the attacker randomly assigns the values to each fake user in the groups g_1 and g_2.

Why is OPA more effective than IPA? By accessing the implementation of the underlying LDP protocols, the attacker can generate and inject poisonous data values that are more effective in affecting the server's final estimation. Specifically, the attacker knows how to solve Equations (4) and (5) by leveraging the knowledge of the LDP perturbation $\Psi()$ and aggregation $\Phi()$. For example, by gaining access to the related parameters, e.g., p, q, k_1, k_2, m_1 and m_2 in $\Psi()$ and $\Phi()$ of SR, the attacker can solve Equations (6) and (7), producing and directly injecting fake values into the output domain of the local LDP instance to launch OPA. As a result, OPA in general will improve the attack performance since the attacker effectively circumvents the LDP perturbation for fake users, thus introducing less noise in the estimation (see the following error analysis).

5 Theoretical Analysis

In this section, we theoretically study the sufficient condition to launch the attack and the attack error given the target mean μ_t and variance σ_t^2. We assume that the user data in g_1 and g_2 have been transformed into $[-1,1]$.

5.1 Sufficient Condition

Sufficient Condition to Launch IPA. The sufficient condition to launch the baseline attack is that Equations (1) and (2) are solvable so that the attacker can find a set of fake input values of the LDP protocol. Specifically, IPA can be launched if the inequalities hold below.

$$-m \le \sum_{i=1}^{m} y_i = (n_e+m)\mu_t - S_e^{(1)} \le m \tag{8}$$

$$y^{2(-)} \le \sum_{i=1}^{m} y_i^2 = (n_e+m)(\sigma_t^2+\mu_t^2) - S_e^{(2)} \le y^{2(+)}, \tag{9}$$

where $y^{2(+)}$ and $y^{2(-)}$ are the maximum and minimum of $\sum_{i=1}^m y_i^2$ under the constraint $\sum_{i=1}^m y_i = (n_e+m)\mu_t - S_e^{(1)}$. Given the input values in the transformed range $[-1,1]$, (8) indicates that the sum of all fake values $\sum_i y_i$ must reside between $-m$ and m when there are m fake users.It is further required in (9) that the sum of the squared fake values $\sum_i y_i^2$ be in the range $[y^{2(-)}, y^{2(+)}]$.

Here we explain how to obtain the above sufficient condition. Since the input value is in the range $[-1,1]$ and there are m fake users, Equation (1) is solvable if $-m \le \sum_{i=1}^m y_i = (n_e+m)\mu_t - S_e^{(1)} \le m$ holds. We then need to determine if Equation (2) is solvable under the constraint $\sum_{i=1}^m y_i = (n_e+m)\mu_t - S_e^{(1)}$. When the range of $\sum_{i=1}^m y_i^2$ under this constraint covers the target σ_t^2, the equation is solvable. To this end, we solve the following optimization problem to find the upper and lower bounds of the term $\sum_{i=1}^m y_i^2$. We first study the maximum of $\sum_{i=1}^m y_i^2$, i.e., the minimum of $-\sum_{i=1}^m y_i^2$.

Theorem 1. *Let $A = (n_e+m)\mu_t - S_e^{(1)}$, when $\lfloor \frac{A+m}{2} \rfloor$ fake values are 1, $m-1-\lfloor \frac{A+m}{2} \rfloor$ fake values are -1 and one fake value is $A - \lfloor \frac{A+m}{2} \rfloor - (m-1-\lfloor \frac{A+m}{2} \rfloor)$, $\sum_{i=1}^m y_i^2$ achieves the maximum.*

Proof. See Appendix A. □

Similarly, we can determine the lower bound of $\sum_{i=1}^m y_i^2$ by changing the objective function from $-\sum_{i=1}^m y_i^2$ to $\sum_{i=1}^m y_i^2$. We omit the detailed steps here but share the result: when all fake values are $\frac{A}{m}$, $\sum_{i=1}^m y_i^2$ achieves the minimum. Given the maximum and minimum of $\sum_{i=1}^m y_i^2$ denoted by $y^{2(+)}$ and $y^{2(-)}$ respectively, we can get the above sufficient condition in (8) and (9).

Sufficient Conditions for OPA. Now we discuss the sufficient conditions for our attack, which will be analyzed in the context of SR and PM respectively.

- *SR mechanism.* The sufficient conditions to launch OPA in SR is that Equations (6) and (7) are solvable so that the attacker can produce viable output for the local LDP instance in order to manipulate the estimate on the server. In SR, the output is either -1 or 1. Therefore, Equations (6) and (7) are solvable if the following hold

$$-m_1 \le (p-q)\left(\frac{n_e+m}{2}\mu_t - \frac{S_e^{(1)}}{2}\right) \le m_1, \tag{10}$$

$$-m_2 \le (p-q)\left(\frac{n_e+m}{2}(\sigma_t^2+\mu_t^2) - \frac{S_e^{(2)}}{2}\right) \le m_2. \tag{11}$$

Inequality (10) indicates that the sum of fake values $\sum_{i=1}^{m_1} \Psi(y_{i,(1)})$ in the output domain should range in $[-m_1, m_1]$ since there are m_1 fake users in g_1. Inequality (11) shows that the range of $\sum_{i=1}^{m_2} \Psi(y_{i,(2)}^2)$ should be from $-m_2$ to m_2 for m_2 fake users in g_2. By estimating m_1 and m_2 to be $\frac{m}{2}$, the attacker obtains the sufficient conditions by determining the value of m that satisfies (10) and (11).

- *PM mechanism.* The analysis of PM is similar to that of SR. Equations (6) and (7) are solvable if the following inequalities hold. We also estimate m_1 and m_2 to be $\frac{m}{2}$.

$$-s \times m_1 \le \frac{n_e+m}{2}\mu_t - S_e^{(1)} \le s \times m_1 \tag{12}$$

$$-s \times m_2 \le \frac{n_e+m}{2}(\sigma_t^2+\mu_t^2) - S_e^{(2)} \le s \times m_2 \tag{13}$$

Since the output of PM is within $[-s,s]$, the corresponding upper and lower bounds in (12) and (13) are multiplied by s.

Number of Fake Users m. The sufficient condition reveals the relationship between the target values and the needed fake user number m, based on which we can further derive the minimum m required to satisfy the sufficient condition given the target mean μ_t and variance σ_t^2. Unfortunately, it is challenging to provide the definite mathematical expression of the minimum m because the inequality signs in the condition

Table 2: Comparison of attack error between baseline and our attack against SR and PM. For a concise comparison, we generate some intermediate notations (e.g., $\mathcal{P}, \mathcal{Q}, \mathcal{T}_{\text{SR}}^{IPA}$, etc.) and show their concrete calculation in Table 5 in Appendix.

	Baseline (IPA)	OPA
$\text{Err}(\hat{\mu}_t)$ in SR	$\mathcal{P} + \frac{2}{(m+n)(p-q)^2} - \mathcal{Q}$	$\frac{\left(2n-2(p-q)^2 S^{(2)}\right)}{(m+n)^2(p-q)^2} + \frac{S^{(2)}}{(m+n)^2} + \mathcal{P}$
$\text{Err}(\hat{\sigma}_t^2)$ in SR	$\leq \frac{2}{(m+n)(p-q)^2} - \frac{S^{(4)}}{(m+n)^2} + \mathcal{T}_{\text{SR}}^{IPA} + 1$	$\leq \frac{2n-2(p-q)^2 S^{(4)}}{(m+n)^2(p-q)^2} + \frac{S^{(4)}}{(m+n)^2} + \mathcal{T}_{\text{SR}}^{OPA} + 1$
$\text{Err}(\hat{\mu}_t)$ in PM	$\frac{2(e^{\varepsilon/2}+3)}{3(n+m)(e^{\varepsilon/2}-1)^2} + \mathcal{P} + \mathcal{Q} + \frac{2\mathcal{Q}}{(e^{\varepsilon/2}-1)}$	$\mathcal{P} + \frac{2n(e^{\varepsilon/2}+3)}{3(m+n)^2(e^{\varepsilon/2}-1)^2} + \frac{(1+e^{\varepsilon/2})S^{(2)}}{(m+n)^2(e^{\varepsilon/2}-1)}$
$\text{Err}(\hat{\sigma}_t^2)$ in PM	$\leq \frac{2(e^{\varepsilon/2}+3)}{3(n+m)(e^{\varepsilon/2}-1)^2} + \frac{2(S^{(4)}+\mathcal{Y}_u^{(4)})}{(n+m)^2(e^{\varepsilon/2}-1)} + \frac{(S^{(4)}+\mathcal{Y}_u^{(4)})}{(m+n)^2} + \mathcal{T}_{\text{PM}}^{IPA} + 1$	$\leq \frac{2n(e^{\varepsilon/2}+3)}{3(m+n)^2(e^{\varepsilon/2}-1)^2} + \frac{(1+e^{\varepsilon/2})S^{(4)}}{(m+n)^2(e^{\varepsilon/2}-1)} + \mathcal{T}_{\text{PM}}^{OPA} + 1$

are uncertain without knowing the values of n_e, $S_e^{(1)}$ and $S_e^{(2)}$. On the other hand, since there are only linear and quadratic terms of m in the inequalities, once n_e, $S_e^{(1)}$, $S_e^{(2)}$, μ_t and σ_t^2 are given, it is not difficult to get the lower bound on m using the quadratic formula. We empirically study the required minimum number of fake users in Section 7.2.4. The results show that a larger m allows the attacker to set the target values farther from the ground truth. In other words, OPA attacker can control fewer fake users but achieve the same attack efficacy as the baseline.

5.2 Error Analysis

In this section, we analyze and compare the attack error of our attack with the baseline attack against SR and PM mechanisms. For ease of analysis, we adopt the widely-used *mean squared error* (MSE) as the error metric. Specifically, let the estimated mean and variance after the attack be $\hat{\mu}_t$ and $\hat{\sigma}_t^2$. Denote $\text{Err}(\hat{\mu}_t)$ and $\text{Err}(\hat{\sigma}_t^2)$ as the MSE $\mathbb{E}[(\hat{\mu}_t - \mu_t)^2]$ and $\mathbb{E}[(\hat{\sigma}_t^2 - \sigma_t)^2]$ between the target and attack result. Our goal is to study $\text{Err}(\hat{\mu}_t)$ and $\text{Err}(\hat{\sigma}_t^2)$.

We summarize the attack error in Table 2 and leave the detailed error analysis with SR to Appendices B and C. We refer readers to the full version [22] for the relevant discussion of PM due to the page limit. Note that it is challenging to solve the exact attack error for the variance because the variance estimation depends on the square of a random variable (i.e., the estimated mean). To address this problem, we consider deriving the upper bound of the attack error for variance.

SR vs. PM We first compare the attack error under different LDP protocols (i.e., SR and PM). We find that the errors are related to the ground truth data due to the terms $S^{(1)}$, $S^{(2)}$ and $S^{(4)}$. By subtracting the error of SR from the error of PM, we find regardless of IPA or OPA, when ε is small, the error under the SR mechanism is smaller than adopting the PM mechanism given a target mean; when ε is large, the attack against PM performs better because PM introduces less LDP error (see Lemma 1). For a target variance, we cannot draw a similar theoretical conclusion because the analysis only provides the upper bound of the error. But our empirical study in Section 7 shows that as ε grows, the error in PM is smaller than in SR due to less LDP noise introduced.

OPA vs. Baseline Now we compare the attack error between OPA and the baseline attack IPA. Theorem 2 shows that regardless of the underlying LDP protocols, OPA outperforms the baseline with less introduced attack error given a target mean; OPA also has a tighter error upper bound with respect to a target variance.

Theorem 2. *The error* $\text{Err}(\hat{\mu}_t)$ *of OPA is smaller than the error* $\text{Err}(\hat{\mu}_t)$ *of IPA, and the upper bound of* $\text{Err}(\hat{\sigma}_t^2)$ *of OPA is smaller than that of IPA.*

Proof. See Appendix D. □

The intuition is that the submitted fake values in IPA will be perturbed by the LDP, which further contributes to the attack errors. However, OPA is able to bypass the perturbation and directly submit fake values to the server. Therefore, LDP noise does not affect the error calculation. However, Theorem 2 only states that OPA has a smaller upper bound of error given a target variance. For completeness, we also empirically study the error. The experimental results show that OPA outperforms the baseline for both target mean and variance.

6 Consistency of Security and Privacy

There is a known security-privacy trade-off in prior research [3, 5], which indicates the incompatible security goal with the privacy requirement of LDP. In other words, prior attacks perform better when ε is set small for higher privacy requirements. However, we do not observe such a trade-off in our proposed data poisoning attack. The security and privacy goals of LDP here are *consistent*, i.e., enhanced privacy also provides improved protection against our data poisoning attack. In this section, we study this consistency for both OPA and IPA, and provide insights into the cause of the difference.

6.1 Security-privacy Consistency in OPA

We analyze the relationship between the attack performance measured by attack error and the privacy level measured by ε and show the result in Theorem 3.

Theorem 3. *For OPA against SR and PM mechanisms, when the privacy budget* ε *is larger, the error on mean and the upper bound of the error on variance become smaller.*

Proof. See Appendix E. □

Theorem 3 only proves that the security-privacy consistency holds for the mean under OPA. The change of the upper bound of the error on variance cannot affirm such consistency result for variance theoretically. Therefore, we also empirically study it and confirm it by our experiments, showing the weakened LDP security as its privacy guarantee deteriorates (see Section 7).

6.2 Which is True: Consistency or Trade-off?

At first glance, the observed security-privacy consistency is at odds with the known result that we have to trade LDP privacy for improved security against unauthorized data injection [3, 5, 47]. Through the foregoing analysis and intuitive reasoning, we discover that the two seemingly conflicting findings actually complement each other. They collectively reveal a more holistic view of the threat landscape in the context of data poisoning attacks. We provide the intuition below.

In general, the relationship between LDP security and its privacy depends on the underlying attack objective. In [5], the goal of the attacker is to impair LDP's overall utility. A small ε facilitates the attack by adding more noise to reduce the accuracy of the result. The constructed false values are independent of the privacy budget for the proposed attack in [3], which aims to maximize the frequency of target items. A small ε allows the fake users to contribute more to the estimated item frequencies, resulting in a higher attack gain. In [47] the security-privacy trade-off remains for the frequency gains of the attack against key-value data [14] since the attack goal is still to maximize the frequency. However, such a trade-off does not necessarily hold when maliciously maximizing the mean. This is because they approximate the mean gain by Taylor expansion in order to perform the attack, which introduces errors into the computation.

Our proposed data poisoning attack has a different goal, i.e., the attacker aims to control the final estimate at a finer level and make the result as close to the target value as possible. Since the attacker can bypass the perturbation and directly inject fake values into the output domain of the local LDP instance, there are two types of errors that impact the result of OPA: the error introduced from the estimation of relevant statistics by the attacker and the error due to the LDP noise from genuine users' input. The former is independent of the LDP privacy implication. For the latter, a small ε incurs large LDP noise such that it is challenging for our attack to precisely manipulate the LDP estimate towards some target value.

The fact that the consistency and trade-off are both valid is disturbing since it complicates the already elusive reasoning and selection of the privacy budget in LDP and makes the design of a secure LDP protocol even more challenging in the presence of different types of data poisoning attacks.

Table 3: Dataset information. The numbers in the parentheses are derived from the original user values before being transformed into $[-1, 1]$.

Dataset	#Sample n	μ	σ^2	$S^{(1)}$	$S^{(2)}$
Taxi [31]	83,130	-0.022 (129)	0.34 (5,932)	-2,194 ($1E7$)	28,362 ($1.8E9$)
Income [35]	2,390,203	-0.93 (51,473)	0.007 ($4.8E9$)	-2,239,154 ($1.2E11$)	2,115,289 ($1.8E16$)
Retirement [32]	97,220	-0.87 (46,249)	0.025 ($3.3E9$)	-85,000 ($4.5E9$)	76,752 ($5.4E14$)

We will discuss the mitigation in Section 8 and the applicability of our attack to other estimations in Section 10.

7 Experiments

7.1 Setup

Dataset. We used three real-world datasets below to evaluate our attack and baseline attack. They all contain numerical values, which we further converted into $[-1, 1]$. More information about the datasets is summarized in Table 3.

- *Taxi* [31]: This dataset comes from 2020 December New York Taxi data, recording the mileage of taxi in a day.
- *Income* [35]: This dataset contains the income of Americans from the 2019 American Community Survey.
- *Retirement* [32]: This dataset contains the salary paid to retired employees in San Francisco.

Metric. We repeatedly run our attacks $N = 100$ times for each evaluation and record the average. We use MSE to measure the attack performance as this metric is widely used for LDP-related evaluations. Let μ_t and σ_t^2 be the target mean and variance, respectively, and the estimated mean and variance in the i-th run be $\hat{\mu}_{t_i}$ and $\hat{\sigma}_{t_i}^2$. Formally, we measure

$$\mathrm{MSE}_\mu = \frac{1}{N}\sum_{i=1}^{N}(\mu_t - \hat{\mu}_{t_i})^2, \ \mathrm{MSE}_\sigma = \frac{1}{N}\sum_{i=1}^{N}\left(\sigma_t^2 - \hat{\sigma}_{t_i}^2\right)^2.$$

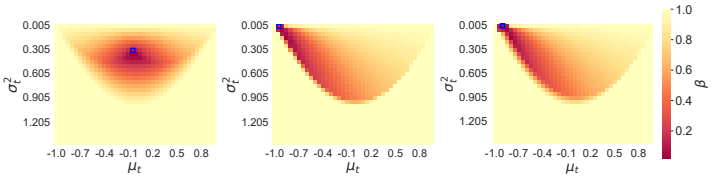
Larger MSE implies worse attack performance since the results are farther from the target values. We did not use error bars because the standard deviation is typically very small and barely noticeable in the figures.

Parameter Setting. We employ a set of default parameters for the evaluation. As shown in Table 4, we have a set of target means $\mu_{t_1}, \mu_{t_2}, \mu_{t_3}$ and a set of target variances $\sigma_{t_1}^2, \sigma_{t_2}^2, \sigma_{t_3}^2$ for each dataset, in which μ_{t_2} and $\sigma_{t_2}^2$ are set to be the true mean and true variance and the rest are randomly produced. We evaluate two cases: 1) the attacker wants to control mean and variance simultaneously by choosing non-true-value targets; 2) she only attempts to manipulate one while keeping the other as is, i.e. the true value.

We choose the default estimated user number n_e^* based on a common observation that online reports tend to publish round numbers instead of precise values [11]. We also use $\beta = \frac{m}{n+m}$

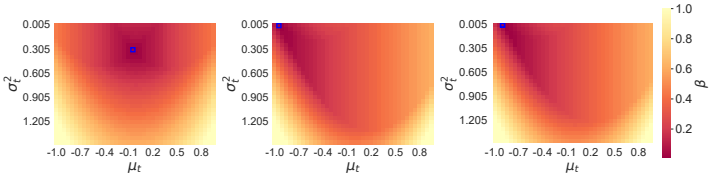
Table 4: Default parameters. Values in parentheses are derived from the original user values before being transformed into $[-1,1]$.

Dataset	μ_{t_1}	μ_{t_2} (true μ)	μ_{t_3}	$\sigma^2_{t_1}$	$\sigma^2_{t_2}$ (true σ^2)	$\sigma^2_{t_3}$	$S_e^{*(1)}$	$S_e^{*(2)}$	n_e^*
Taxi	0.06 (140)	-0.022 (129)	-0.06 (118)	0.33 (5,793)	0.34 (5,932)	0.4 (7,022)	-2,326 (1E7)	29,580 (1.9E9)	80,000
Income	-0.92 (65,160)	-0.93 (51,473)	-0.94 (48,870)	0.005 (3.3E9)	0.007 (4.8E9)	0.009 (5.9E9)	-2,234,086 (1.2E11)	2,108,453 (2E16)	2,400,000
Retirement	-0.86 (51,515)	-0.87 (46,249)	-0.88 (44,156)	0.02 (2.7E9)	0.025 (3.3E9)	0.03 (4E9)	-85,157 (4.4E9)	77,032 (5.3E14)	100,000



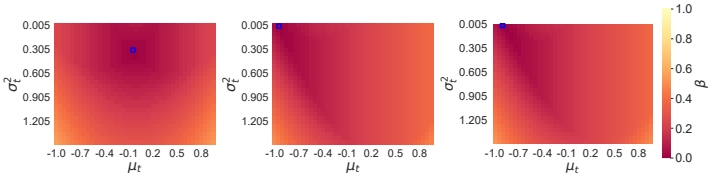
(a) Taxi (b) Income (c) Retirement

Figure 2: The minimum number of fake users needed for IPA with $\varepsilon = 1$ and varying μ_t and σ_t^2 independent of SR and PM.



(a) Taxi (b) Income (c) Retirement

Figure 3: The minimum number of fake users to launch OPA on SR with $\varepsilon = 1$ varying μ_t and σ_t^2.



(a) Taxi (b) Income (c) Retirement

Figure 4: The minimum number of fake users needed for OPA on PM with $\varepsilon = 1$ and varying μ_t and σ_t^2.

to denote the ratio of the number of fake users to the number of total users and set the default $\beta = 0.1$. $S^{(1)}$ and $S^{(2)}$ can be estimated by either getting access to recent published historical data or compromising a small group of users. Without loss of generality, we randomly select $1,000$ users in each dataset to gain the required information to simulate the public data source and they remain as genuine users in our experiment.

7.2 Results

Here we report the experimental results of the sufficient condition evaluation and impact of various factors on the attack.

7.2.1 Sufficient Condition for Achieving Target Values

We first study the sufficient condition for IPA and OPA to achieve the μ_t and σ_t^2 on SR and PM. Specifically, given the attacker-estimated $S_e^{(1)}$, $S_e^{(2)}$ and n_e, we study the relationship between μ_t and σ_t^2 and the minimum number of fake users (measured by $\beta = \frac{m}{n+m}$) required to launch the attack. The results are shown in Figure 2, 3 and Figure 4. Since the baseline IPA is independent of LDP, the required minimum number of fake users is the same for both SR and PM. Thus, we use one plot for each dataset regardless of the underlying LDP protocols in Figure 2. OPA leverages the underlying LDP mechanisms, i.e. SR in Figure 3 and PM in Figure 4. We highlight the true mean and variance in the encoded range $[-1,1]$ with a blue rectangle, and use the darker color to indicate that fewer fake users are needed for a successful attack. The light-color parts ($\beta = 1$) represent the extreme cases where the attack is infeasible given a pair of μ_t and σ_t^2. We can derive the following key observations:

- In general, more fake users allows the attacker in both baseline and our attack to set a target value that deviates farther from the ground true. β keeps small when μ_t and σ_t^2 grow simultaneously. This is because when μ_t and σ_t^2 grow, both constraint terms for $\sum_i^m y_i$ and $\sum_i^m y_i^2$ (in Equations (1) and (2)) increase together and thus a small number of large fake values can satisfy the constraints.
- OPA can reach more darker regions compared to the baseline attack, which means our attacker has a wider selection of target mean and variance pairs and can successfully accomplish the attack given the selected targets. This is because OPA attacker can inject fake values into the LDP output domain. Compared with IPA, the constraints of the fake values are relaxed by the factors in the LDP aggregation (Section 5.1), thus fewer fake users needed for OPA.
- For OPA, the accessible region of target values in SR is smaller than in PM. This is because the factors in SR aggregation are smaller than those in PM, leading to a set of tighter constraints for fake values and thus the target values spreading over a smaller region.

7.2.2 Impact of Target Values

Figure 5 and 6 depict the attack performance against SR and PM with varying target mean and variance. We observe that

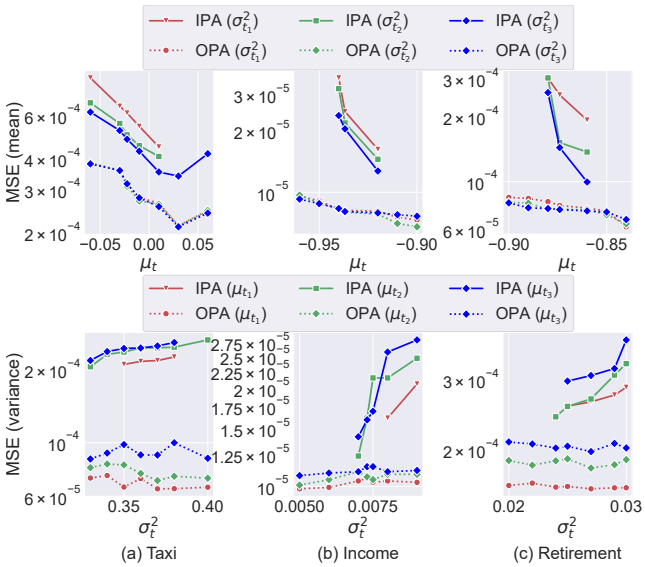


Figure 5: Attack error in SR, varying μ_t and σ_t^2. Set $\varepsilon = 1$, $m = 0.1n$, $S_e^{(1)} = S_e^{*(1)}$ and $S_e^{(2)} = S_e^{*(2)}$, $n_e = n_e^*$.

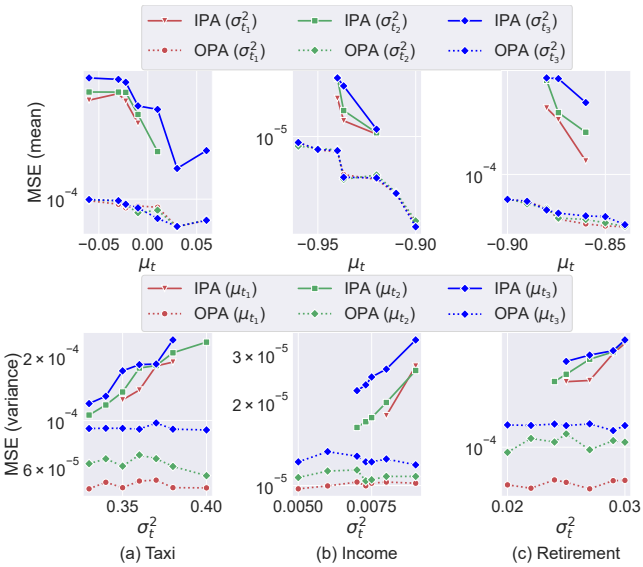


Figure 6: Attack error in PM, varying μ_t and σ_t^2. Set $\varepsilon = 1$, $m = 0.1n$, $S_e^{(1)} = S_e^{*(1)}$ and $S_e^{(2)} = S_e^{*(2)}$, $n_e = n_e^*$.

- OPA outperforms IPA when attacking the mean, since OPA circumvents LDP perturbation on the fake user side, resulting in less attack error. For example, on *Taxi*, OPA reduces MSE of IPA by about 30% on SR and PM respectively.

- OPA also outperforms IPA when attacking the variance with all three datasets. As the target variance grows, the MSE of the baseline IPA against both SR and PM increases. This is because when the target increases, the bias in the attack error grows in the SR mechanism, and both variance and bias increase in PM. From Figure 5, we observe a much reduced MSE with OPA, e.g., at most 50% error reduction compared to the baseline against SR by controlling the same number of fake users.

- For some target mean/variance, the default β is inadequate to launch the baseline attack. Thus, no corresponding MSE is recorded within the given range in the figures. The result again exhibits the advantage of OPA over the baseline when the attacker can only control a limited number of users.

- Given a target μ_t, the MSE of IPA becomes smaller (larger) on SR (on PM) as σ_t^2 grows. This is because a larger σ_t^2 provides a larger $\sum_i y_i^2$, thus leading to small error in SR and larger error in PM. However, OPA errors are not affected by $\sum_i y_i^2$ and are close under different σ_t^2.

- The MSE of IPA($\sigma_{t_3}^2$) and OPA($\sigma_{t_3}^2$) on *Taxi* reduces first then increases as μ_t grows. This is because both MSE of IPA and OPA are a quadratic function of μ_t (see error analysis in Section 5.2). On *Income* and *Retirement*, the range of μ_t only covers the reduction part of quadratic MSE. Figure 5 and 6 thus do not show a parabola shape.

To have a more intuitive comparison with the target, we observe that the OPA result deviates from the target mean by as small as 0.22% in the figures versus 0.58% with IPA and a maximum error reduction from 17% with IPA to only 3.7% with OPA given a target variance.

7.2.3 Impact of Attacker's Estimation

We study how the relevant estimation of the attacker affects the results next. In general, more accurate estimation pushes the attack result further to the target.

Impact of $S_e^{(1)}$ and $S_e^{(2)}$. We discuss the impacts of $S_e^{(1)}$ (Figure 7) and $S_e^{(2)}$ (Figure 8) on attack performance. Note that the intervals in the x-axis of both figures are different across the datasets because the respective $S^{(1)}$ and $S^{(2)}$ are distinct. Below are some key observations.

- Figures 7 and 8 show that OPA outperforms IPA when attacking mean and variance across all datasets.

- The more accurate the estimation of $S_e^{(1)}$ is, the smaller the attack error is as shown in Figure 7. The MSEs of both OPA and baseline are approximately symmetrical about the estimation error $S^{(1)} - S_e^{(1)} = 0$, because the term $(S^{(1)} - S_e^{(1)})^2$ in the attack error computation increases when $S_e^{(1)}$ moves farther away from $S^{(1)}$ (see Table 2).

- In Figure 8, we observe the similar impact of $S_e^{(2)}$ on variance. However, the impact on the mean shows a different trend. With increased $S_e^{(2)}$, the MSE of IPA against SR grows but reduces against PM. This is reasonable because when $S_e^{(2)}$ grows, the sum of the squared fake values $\sum_{i=1}^{m} y_i^2$ decreases, leading to a large error in SR and a small error in PM (Lemma 1). The MSE of OPA remains almost constant with varying $S_e^{(2)}$ because the attacker crafts fake values in g_1 to manipulate the mean, which is not affected by $S^{(2)}$.

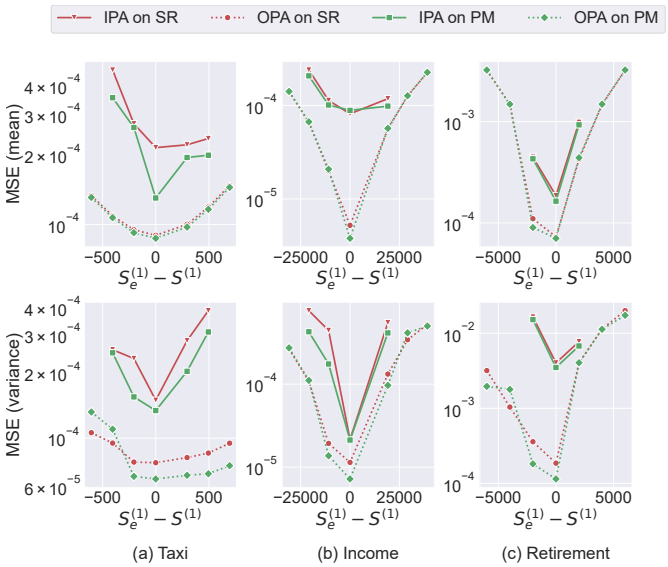


Figure 7: Attack error in SR and PM, varying $S_e^{(1)}$. Target values are μ_{t_1} and $\sigma^2_{t_1}$, $\varepsilon = 1$, $m = 0.1n$, $S_e^{(2)} = S_e^{*(2)}$, $n_e = n_e^*$.

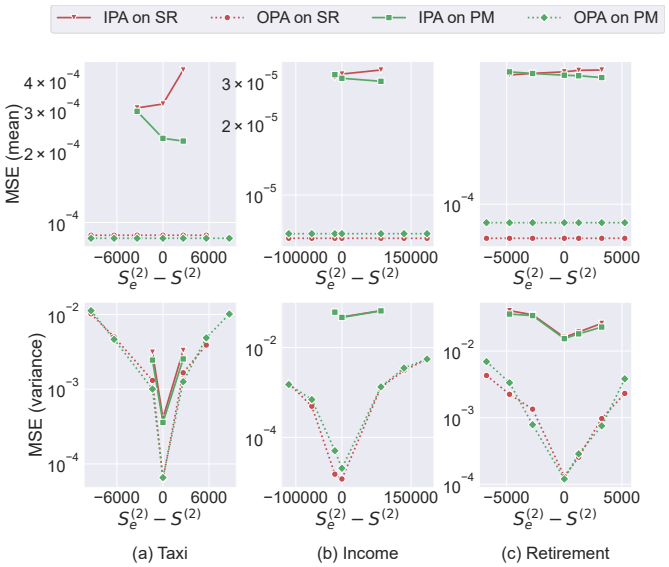


Figure 8: Attack error in SR and PM, varying $S_e^{(2)}$. Target values are μ_{t_1} and $\sigma^2_{t_1}$, $\varepsilon = 1$, $m = 0.1n$, $S_e^{(1)} = S_e^{*(1)}$, $n_e = n_e^*$.

- Due to the constrained sufficient condition for launching IPA, the default β is insufficient when both $S_e^{(1)}$ and $S_e^{(2)}$ are far away from $S^{(1)}$ and $S^{(2)}$ respectively, thus no corresponding MSE recorded in Figure 7 and 8.

The recovered value on the server gets closer to the target with a smaller estimation error. In practice, the interpretation of the attack efficacy is subject to the attacker and may vary depending on applications. For example, we in the experiment observe that when an OPA attacker sets the target mean to 0.06 and target variance to 0.33 on *Taxi* with 18% estimation error about $S_e^{(1)}$, the recovered mean and variance by SR are 0.056 (about 3% from the target) and 0.329 (about 0.9% from the target) respectively, which may still be considered a success by the attacker.

Impact of n_e. We can derive a similar conclusion regarding the relationship between estimation error $n_e - n$ and attack error in Figure 9. In general, less estimation error results in better attack performance. The MSEs of IPA and OPA are almost symmetric about $n_e - n$. Besides, OPA performs better in both SR and PM due to more LDP noise introduced in IPA.

Similar to estimating $S_e^{(1)}$ as analyzed previously, the attacker may not be able to get an accurate estimate of the user number in practice, which will cause the recovered statistics to deviate from the intended values. Again, the interpretation of the deviation here is subject to the attacker's objective. Our experiment reports that given the target mean -0.86, variance 0.02, 10% estimation error of user number (88,000 estimated vs. 97,220 actual) on *Retirement*, a server using SR mechanism can recover the mean and variance to -0.861 (about 0.36% accuracy loss) and 0.0202 (about 4% accuracy loss) respectively, under our output poisoning attack.

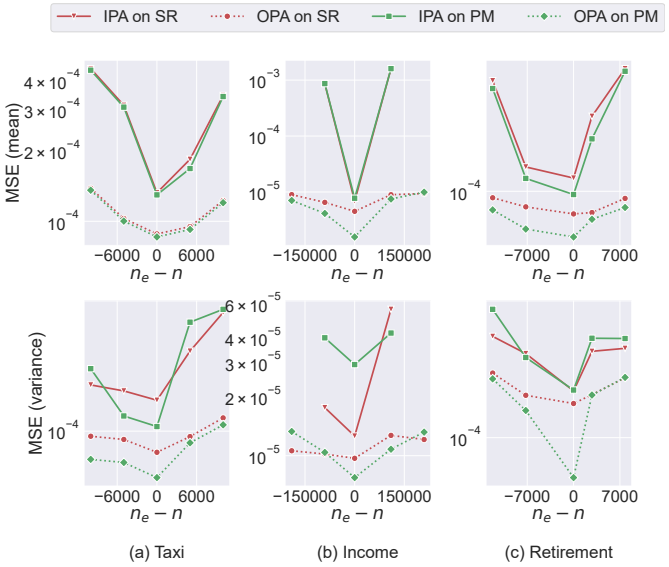


Figure 9: Attack error in SR, varying n_e. Target values are μ_{t_1} and $\sigma^2_{t_1}$, $\varepsilon = 1$, $m = 0.1n$, $S_e^{(1)} = S_e^{*(1)}$, $S_e^{(2)} = S_e^{*(2)}$.

7.2.4 Other Factors

We further study the impact of the ε, β and time cost for the attack. The results confirm the better performance of OPA, the security-privacy consistency and the attack efficiency.

Impact of ε. Figure 10 shows how ε affects attack performance. We empirically confirm the privacy-security consistency with our attacks, which complements theoretical analysis in Section 6. Overall, the attack performance improves with large ε. For the attack on mean and variance, OPA exceeds IPA under all selected ε since OPA is partially influenced by LDP obfuscation. As ε increases, the attack error in PM

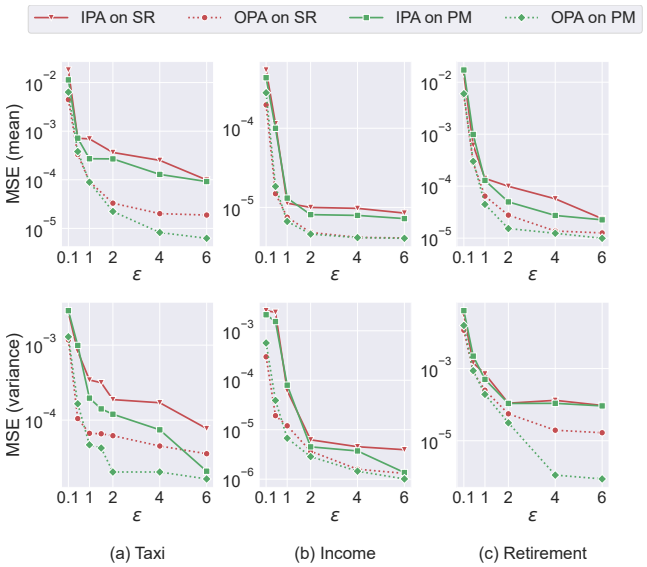


Figure 10: Attack error in SR and PM, varying ε. Target values are μ_{t_1} and $\sigma^2_{t_1}$, $m = 0.1n$, $S_e^{(1)} = S_e^{*(1)}$, $S_e^{(2)} = S_e^{*(2)}$, $n_e = n_e^*$.

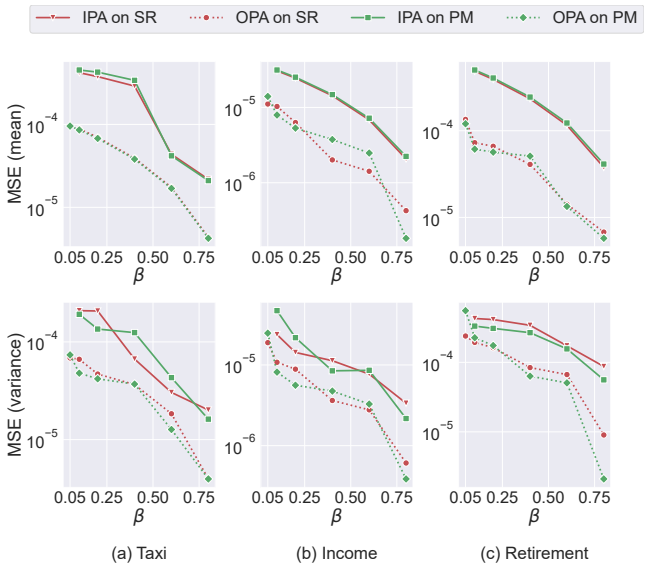


Figure 11: Attack error in SR and PM, varying β. Target values are μ_{t_1} and $\sigma^2_{t_1}$, $\varepsilon = 1$, $S_e^{(1)} = S_e^{*(1)}$, $S_e^{(2)} = S_e^{*(2)}$, $n_e = n_e^*$

is smaller than that in SR because PM adds less LDP noise (see Lemma 1). However, even when $\varepsilon = 0.1$, the OPA results are still close to the targets, i.e. the accuracy loss is less than 2.2% for the target mean and 4% for the target variance on *Retirement*.

Impact of β. Figure 11 shows the impact of the number of fake users. In reality, an appropriate β depends on the application and the resources accessible to the attacker. We varied β from 0.05 to 0.8 to comprehensively evaluate the attack by covering extreme cases. We observe that the attack errors of both baseline and OPA on mean and variance reduce as β grows. This is because the number of fake users is in the denominator of the error calculation for both IPA and OPA. However, OPA performs much better since OPA is only partially affected by LDP noise. For the default $\beta = 0.1$ on *Retirement*, the accuracy loss of OPA toward target mean and variance is only about 0.03% and 0.93% respectively.

Time Efficiency. If the required estimates have been done in advance, the time cost for the attacks only depends on the calculation of the fake values to be injected. Per our experiment, performing OPA is faster (i.e. less than 0.2 seconds) than IPA (about 10 seconds). This is because the OPA can directly produce fake values given the explicit expression of $\Psi(y)$ and $\Psi(y^2)$ while the optimization problem in Equation (3) needs to be solved for IPA. We adopted PyTorch in our experiment. Other optimizers may lead to different results for IPA.

8 Mitigation

There are two types of methods proposed in prior research to defend against the data poisoning attack, i.e. normalization [3] and fake user detection [3, 47]. The idea of normalizing the LDP estimates to reduce attack effectiveness is based on the naïve observation of frequency consistency [44], which is not applicable to mean/variance estimation. Detecting fake users is possible if the fake values are distinguishable from normal values. We present a countermeasure that can tolerate the skewed values and recover the true estimate. Contrary to prior work that assumes the server knows user values and the fraction of genuine users as ground truth [47], we consider these conditions are difficult to satisfy in reality and our defense does not rely on them.

8.1 Clustering-based Defense

We adopt a sampling-then-clustering method to defend against our output poisoning attack, inspired by [4] in the context of federated learning. The main idea is to sample multiple subsets of users and then use a clustering algorithm, such as k-means, to form two clusters. The cluster that contains more subsets will be used for estimation, while the other will be discarded. The intuition is that since the majority of users are genuine, the mean of most subsets should be similar and close to the true mean. More precisely, we first define a sampling rate r $(0 < r < 1)$ and derive all $\binom{n_1}{rn_1}$ possible subsets in g_1 and all $\binom{n_2}{rn_2}$ possible subsets in g_2 without replacement, where n_i is the number of users in g_i. Next, we estimate $\mathbb{E}(x)$ and $\mathbb{E}(x^2)$ for each subset and feed them into k-means for g_1 and g_2. By identifying the benign clusters in g_1 and g_2, we use their respective cluster centers as $\mathbb{E}(x)$ or $\mathbb{E}(x^2)$ for mean and variance estimation. Our defense could be further optimized by leveraging advanced fault tolerance results [15, 24], which will be left as an important future work.

Results. We evaluate the defense performance by measuring

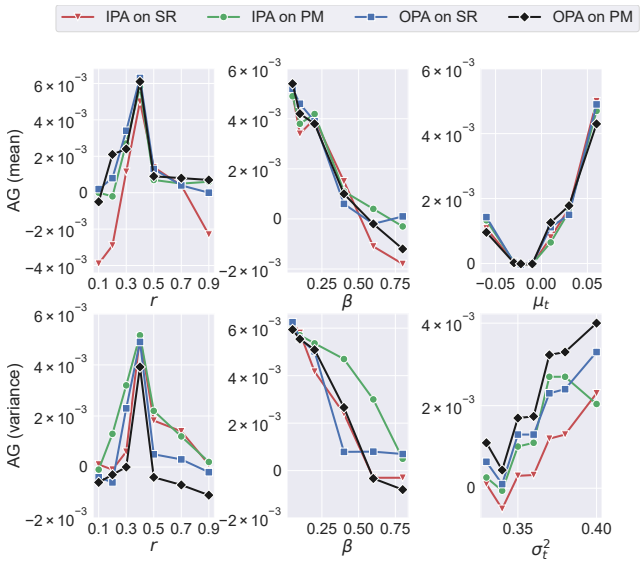


Figure 12: Defense evaluation results. True $\mu = -0.22$ and $\sigma^2 = 0.34$. The default target values are μ_{t_1} and $\sigma^2_{t_1}$, $\varepsilon = 1$, $\beta = 0.1$, $S_e^{(1)} = S_e^{*(1)}$, $S_e^{(2)} = S_e^{*(2)}$, $n_e = n_e^*$.

the *accuracy gain* (AG) after applying our mitigation. AG measures the estimation error change before and after the proposed defense as $MSE_{before} - MSE_{after}$. A positive AG value indicates the defense helps the LDP regain the data accuracy after the attack while a negative one represents the ineffectiveness of the mitigation. Thus, the larger AG is, the more effective our defense is against the attack. In practice, the number of subsets $\binom{n_i}{rn_i}$ could be too large for an efficient response. We observe that further randomly choosing a small portion (e.g. 5,000 subsets in our experiment) still results in an effective defense.

We use the dataset *Taxi* for result demonstration in Figure 12. It shows that the choice of r will affect the performance (the leftmost figures). A small sampling rate will lead to a small subset, which in turn introduces more bias. On the other hand, a large r results in fewer subsets, but each subset may contain more fake users, thus being subject to manipulation. This may explain why negative AG is observed when r is chosen either too small or large. We empirically find an optimal r for the rest of the evaluation. The defense performance is also related to the ratio β (the middle figures). When the fraction of fake users is small, our defense is very effective. When the target value is far from the true value, it is easier to identify the fake values as outliers with increased AG (the rightmost figures). Therefore, our method is preferred when the fraction of fake users is small and the attacker wants to skew the mean and variance substantially, which is reasonable for most attacking scenarios.

8.2 Other Defenses

For an attack that aims to falsify the input of a local LDP instance, such as the baseline IPA, an authenticated data feed system may assist in reestablishing trust for data sources. The current solution, however, is limited to well-known entities [49]. Authenticating data from unidentified sources in a distributed environment remains an open problem [19, 49]. To defend against data poisoning attacks for frequency and heavy hitter estimations, two cryptography-based methods were proposed in [19] and [29] respectively. Kato *et al.* [19] utilized cryptographic randomized response as a building block to make the LDP protocol for frequency estimation verifiable to the data curator. In [29], the multi-party computation was also leveraged to restrict the attacker's capability to manipulate the heavy hitter results. These countermeasures could be used to mitigate the data poisoning attacks in [3, 47], but are not applicable to our attacks due to the different LDP perturbation mechanisms for mean and variance estimations. Other potential solutions include adopting hardware-assisted trusted computing [1, 21] to ensure the authenticity of the LDP protocol execution and communication with the remote server. But this may incur additional costs for software-hardware co-design and security auditing in order to avoid a wide spectrum of side-channel attacks [13, 28, 37, 40, 46].

9 Related Work

Data Poisoning Attack to LDP Protocols. Recent research found that LDP is vulnerable to data poisoning attacks. It was shown that the LDP aggregator is sensitive to distribution change of perturbed data. Thus, the result accuracy of non-interactive LDP protocols can be degraded by injecting false data [5]. It is further demonstrated that the attacker can maximally deviate the estimate of an item of interest (e.g. categorical data in [3] and key-value data in [47]) from the ground truth by formulating the attack as an optimization problem to maximize the attack gain. The solution is the fake data that the attacker will send to the data collector.

We consider a fine-grained attack in this paper, where the attacker aims to control the estimate to some desired values. In general, the above maximal deviation attacks [3, 47] can be deemed as the extreme cases of our attack. With this new capability, the attacker can target more scenarios for precise result manipulation. In addition, we provide important new insights into the attack impact on LDP and mitigation design.

Adopting DP/LDP against Poisoning Attacks in ML. A line of work [2, 25, 30] studied using DP/LDP to improve the robustness of machine learning models against data poisoning attacks, where the attacker prepares a poisoned training dataset to change the model behavior to some desired one. In [2], differentially private data augmentation was studied for attack mitigation. [25] investigated the attacks on the ML model trained with differentially private algorithms. Other than central DP, [30] further studied the impact of LDP on the defense and observed varying levels of protection-utility trade-offs. Our work has a distinct motivation, i.e., we study

the intrinsic robustness and security of LDP protocols in the presence of fine-grained data manipulation. The results of our work may shed light on the related security discussions of using LDP for defenses. For example, a strong LDP perturbation may help reinforce the defense effect in [30] while the attack in a central DP setting [2, 25, 30] is intuitively analogous to our baseline attack where only input values can be crafted.

10 Discussion

There exist other LDP protocols supporting mean/variance estimation [9, 23, 42], to which the baseline and our attack are still applicable. IPA is straightforward since it is independent of LDP implementation. For OPA, the attacker can craft fake data in the output domain of the perturbation by leveraging the LDP knowledge. Note that since the aggregation $\Phi()$ is iterative in [23], we cannot derive an explicit mathematical expression to determine fake values in the same way as in this work (e.g., Equations (6) and (7)). However, the attacker may obtain a valid solution by simulating the iteration and searching the output domain of the perturbation.

Frequency Estimation under Pure LDP [42]: The IPA and OPA could be adapted to attack the pure LDP protocols for frequency, such as kRR [7], OUE and OLH [42]. The attacker needs to estimate the frequencies of items and inject bogus data as per the the intended frequency of target items. OPA may leverage the LDP protocols to improve performance.

Distribution Estimation [23]: Distribution estimation can be considered a frequency oracle in the numerical domain, to which our attacks may still be applicable. We provide the attack intuition here. In general, the attacker begins by estimating the original data distribution. Given this, the attack may generate fake data points equal to a specific value $\boldsymbol{x}$ to increase the probability density of $\boldsymbol{x}$ to the target value. To reduce the probability density, the attacker could provide data that is not equal to $\boldsymbol{x}$.

Graph data mining [16, 48]: In graph data mining, LDP protocols focus on calculating graph statistics, e.g., counting triangles and k-stars in the graph, the degree and adjacency bit vector of each node. We assume in this scenario that the attacker wishes to control the final estimate to some target value. To launch the attack, the attacker could first use a graph generation model, such as BTER [34], to estimate the graph topology. The attacker then could inject bogus nodes and edges into the graph to exert finer control over its statistics.

For all the discussed query types, the security-privacy consistency may remain, as increased privacy introduces additional noise and reduces the effectiveness of the manipulation.

11 Conclusion

We conducted a systematic study on data poisoning attacks against the LDP protocols for mean and variance estimation. We present an effective attack to craft the output of the LDP instance and manipulate both mean and variance estimates according to target values. The analysis reveals a disturbing fact: the LDP is inherently vulnerable to data poisoning attacks regardless of the privacy budget, i.e., previous attacks are effective when ε is small (high privacy), whereas our attacks perform better when ε is large (low privacy). We also discussed the applicability of our attacks against other query types and shed light on the promising mitigation development.

12 Acknowledgment

We would like to thank the Shepherd and anonymous reviewers for their insightful comments and guidance. Hui Li and Xiaoguang Li were supported in part by NSFC 61932015.

References

[1] ARM. Arm confidential compute architecture. https://developer.arm.com/architectures/architecture-security-features, 2021. Accessed on Mar 31, 2021.

[2] Eitan Borgnia, Jonas Geiping, Valeriia Cherepanova, Liam Fowl, Arjun Gupta, Amin Ghiasi, Furong Huang, Micah Goldblum, and Tom Goldstein. Dp-instahide: Provably defusing poisoning and backdoor attacks with differentially private data augmentations. *arXiv preprint arXiv:2103.02079*, 2021.

[3] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Data poisoning attacks to local differential privacy protocols. In *USENIX Security*, 2021.

[4] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Provably secure federated learning against malicious clients. In *AAAI*, 2021.

[5] Albert Cheu, Adam Smith, and Jonathan Ullman. Manipulation attacks in local differential privacy. In *IEEE S&P*, 2021.

[6] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. *arXiv preprint arXiv:1712.01524*, 2017.

[7] John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *IEEE FOCS*, 2013.

[8] John C Duchi, Michael I Jordan, and Martin J Wainwright. Minimax optimal procedures for locally private estimation. *Journal of the American Statistical Association*, 113(521):182–201, 2018.

[9] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[10] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[11] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *ACM CCS*, 2014.

[12] Kayla Fontenot, Jessica Semega, Melissa Kollar, et al. Income and poverty in the united states: 2017. *Washington DC: US Government Printing Office*, 2018.

[13] Johannes Götzfried, Moritz Eckert, Sebastian Schinzel, and Tilo Müller. Cache attacks on intel SGX. In *Proceedings of the 10th European Workshop on Systems Security*, 2017.

[14] Xiaolan Gu, Ming Li, Yueqiang Cheng, Li Xiong, and Yang Cao. PCKV: Locally differentially private correlated key-value data collection with optimized utility. In *USENIX Security*, 2020.

[15] Nirupam Gupta and Nitin H Vaidya. Fault-tolerance in distributed optimization: The case of redundancy. In *ACM PODC*, 2020.

[16] Jacob Imola, Takao Murakami, and Kamalika Chaudhuri. Locally differentially private analysis of graph statistics. In *USENIX Security*, 2021.

[17] Jingdong Big Data Research. White paper on consumption of the post-90s generation. http://pdf.dfcfw.com/pdf/H3_AP202009111410915927_1.pdf, 2020. Accessed on Jan 10, 2021.

[18] William Karush. Minima of functions of several variables with inequalities as side conditions. In *Traces and Emergence of Nonlinear Programming*. Springer, 2014.

[19] Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. Preventing manipulation attack in local differential privacy using verifiable randomization mechanism. *arXiv preprint arXiv:2104.06569*, 2021.

[20] Harold W Kuhn and Albert W Tucker. Nonlinear programming. In *Traces and emergence of nonlinear programming*. Springer, 2014.

[21] Dayeol Lee, David Kohlbrenner, Shweta Shinde, Krste Asanović, and Dawn Song. Keystone: An open framework for architecting trusted execution environments. In *EuroSys*, 2020.

[22] Xiaoguang Li, Ninghui Li, Wenhai Sun, Neil Zhenqiang Gong, and Hui Li. Fine-grained poisoning attack to local differential privacy protocols for mean and variance estimation. *arXiv preprint arXiv:2205.11782*, 2022.

[23] Zitao Li, Tianhao Wang, Milan Lopuhaä-Zwakenberg, Ninghui Li, and Boris Škoric. Estimating numerical distributions under local differential privacy. In *ACM SIGMOD*, 2020.

[24] Shuo Liu, Nirupam Gupta, and Nitin H Vaidya. Approximate byzantine fault-tolerance in distributed optimization. In *ACM PODC*, 2021.

[25] Yuzhe Ma, Xiaojin Zhu, and Justin Hsu. Data poisoning against differentially-private learners: Attacks and defenses. *arXiv preprint arXiv:1903.09860*, 2019.

[26] Microsoft. Microsoft: By the numbers. https://news.microsoft.com/bythenumbers/en/windowsdevices, 2022. Accessed on Aug 10, 2022.

[27] Takao Murakami and Yusuke Kawamoto. Utility-optimized local differential privacy mechanisms for distribution estimation. In *USENIX Security*, 2019.

[28] Kit Murdock, David Oswald, Flavio D Garcia, Jo Van Bulck, Daniel Gruss, and Frank Piessens. Plundervolt: Software-based fault injection attacks against Intel SGX. In *IEEE S&P*, 2020.

[29] Moni Naor, Benny Pinkas, and Eyal Ronen. How to (not) share a password: Privacy preserving protocols for finding heavy hitters with adversarial behavior. In *ACM CCS*, 2019.

[30] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. Local and central differential privacy for robustness and privacy in federated learning. In *NDSS*, 2022.

[31] NYC Taxi and Limousine Commission. TLC trip record data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page, 2018. Accessed on May 10, 2021.

[32] S.F.C. Office. SF employee compensation. https://www.kaggle.com/san-francisco/sf-employee-compensation#employee-compensation.csv, 2019. Accessed on May 10, 2021.

[33] Opacus. Pytorch privacy. https://github.com/ebagdasa/pytorch-privacy, 2019. Accessed on July 10, 2021.

[34] Zhan Qin, Ting Yu, Yin Yang, Issa Khalil, Xiaokui Xiao, and Kui Ren. Generating synthetic decentralized social graphs with local differential privacy. In *ACM CCS*, 2017.

[35] Steven Ruggles, Sarah Flood, Ronald Goeken, Josiah Grover, Erin Meyer, Jose Pacas, and Matthew Sobek. Integrated public use microdata series: Version 9.0 [database]. Minneapolis: University of Minnesota. http://doi.org/10.18128/D010.V9.0, 2019. Accessed on May 10, 2021.

[36] Ellis Scharfenaker, Markus Schneider, et al. *Labor market segmentation and the distribution of income: New evidence from internal census bureau data*. University of Utah, Department of Economics, 2019.

[37] Michael Schwarz, Samuel Weiser, Daniel Gruss, Clémentine Maurice, and Stefan Mangard. Malware guard extension: Using SGX to conceal cache attacks. In *DIMVA*, 2017.

[38] Emily Shrider, Melissa Kollar, Frances Chen, and Jessica Semega. Income and poverty in the united states: 2020. September, 2021.

[39] Statista. Twitter - Statistics & Facts. https://www.statista.com/topics/737/twitter/#dossierKeyfigures, 2021. Accessed on Oct 1, 2021.

[40] Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F Wenisch, Yuval Yarom, and Raoul Strackx. Foreshadow: extracting the keys to the Intel SGX kingdom with transient out-of-order execution. In *USENIX Security*, 2018.

[41] Ning Wang, Xiaokui Xiao, Yin Yang, Jun Zhao, Siu Cheung Hui, Hyejin Shin, Junbum Shin, and Ge Yu. Collecting and analyzing multidimensional data with local differential privacy. In *IEEE ICDE*, 2019.

[42] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Locally differentially private protocols for frequency estimation. In *USENIX Security*, 2017.

[43] Tianhao Wang, Bolin Ding, Jingren Zhou, Cheng Hong, Zhicong Huang, Ninghui Li, and Somesh Jha. Answering multi-dimensional analytical queries under local differential privacy. In *ACM SIGMOD*, 2019.

[44] Tianhao Wang, Milan Lopuhaa-Zwakenberg, Zitao Li, Boris Skoric, and Ninghui Li. Locally differentially private frequency estimation with consistency. In *NDSS*, 2020.

[45] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[46] Nico Weichbrodt, Anil Kurmus, Peter Pietzuch, and Rüdiger Kapitza. Asyncshock: Exploiting synchronisation bugs in intel SGX enclaves. In *ESORICS*, 2016.

[47] Yongji Wu, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Poisoning attacks to local differential privacy protocols for key-value data. In *USENIX Security*, 2022.

[48] Qingqing Ye, Haibo Hu, Man Ho Au, Xiaofeng Meng, and Xiaokui Xiao. Towards locally differentially private generic graph metric estimation. In *IEEE ICDE*, 2020.

[49] Fan Zhang, Ethan Cecchetti, Kyle Croman, Ari Juels, and Elaine Shi. Town crier: An authenticated data feed for smart contracts. In *ACM CCS*, 2016.

A Proof of Theorem 1

In the optimization problem, the objective function $-\sum_{i=1}^{m} y_i^2$ is convex, the inequality constraints are continuously differentiable convex functions and the equality constraint is an affine function. Thus we can prove that the solution is the maximum by proving the solution satisfies KKT conditions [18, 20]. We prove a general case where the value is in $[a,b]$, and Theorem 1 can be derived by setting $a=-1$ and $b=1$. Let the function $L(y_1,...,y_m)$ equal

$$-\sum_{i=1}^{m} y_i^2+\alpha(\sum_{i=1}^{m} y_i-A)+\sum_{i=1}^{m}\left[\beta_i^{(a)}(a-y_i)+\beta_i^{(b)}(y_i-b)\right],$$

where $A=(n+m)\mu_t-S_e^{(1)}$, and α, $\beta_i^{(a)}$ and $\beta_i^{(b)}$ are constants. The solution should satisfy four sets of conditions:

Stationarity: $\forall i: \frac{\partial L}{\partial y_i}=-2y_i+\alpha-\beta_i^{(a)}+\beta_i^{(b)}=0$.

Primal feasibility: $\sum_{i=1}^{m} y_i-A=0, \forall i: a-y_i\le 0, y_i-b\le 0$.

Dual feasibility: For any i, $\beta_i^{(a)},\beta_i^{(b)}\ge 0$.

Complementary slackness: $\forall i: \beta_i^{(a)}(a-y_i)=0, \beta_i^{(b)}(y_i-b)=0$.

Since the partial derivative of L should be zero, we have

$$y_i=\frac{1}{2}(\alpha-\beta_i^{(a)}+\beta_i^{(b)}) \Rightarrow \frac{1}{2}\sum_{i}^{m}(\alpha-\beta_i^{(a)}+\beta_i^{(b)})=A.$$

Thus we can rewrite α and y_i as $\alpha=\frac{2A+\sum_i^m\beta_i^{(a)}-\sum_i^m\beta_i^{(b)}}{m}$ and $y_i=\frac{1}{2}(\frac{2A+\sum_i^m\beta_i^{(a)}-\sum_i^m\beta_i^{(b)}}{m}-\beta_i^{(a)}+\beta_i^{(b)})$. Let the domain of y_i be $D=D_a\bigcup D_b\bigcup D_{ab}$ s.t., $\forall y_i\in D_a, y_i=a$, $\forall y_i\in D_b, y_i=b$, and $\forall y_i\in D_{ab}, a<y_i<b$. Given the solution, we have $|D_b|=\lfloor\frac{A-ma}{b-a}\rfloor$, $|D_a|=m-1-|D_b|$ and $|D_{ab}|=1$. For $\forall y_i\in D_a$, we have $\forall i:\beta_i^{(b)}=0$ due to the complementary slackness, and

$$y_i=a=\frac{1}{2}(\frac{2A+\sum_i^m\beta_i^{(a)}}{m}-\beta_i^{(a)})$$
$$\Rightarrow \beta_i^{(a)}=\frac{2A+(m-1-|D_b|)\beta_i^{(a)}}{m}-2a=\frac{2(A-ma)}{|D_b|-1}.$$

Since $A\ge ma$ and $|D_b|\ge 1$, we have $\beta_i^{(a)}\ge 0$ for all $y_i\in D_a$. Therefore, for $\forall i: y_i\in D_a$, we have $\beta_i^{(a)},\beta_i^{(b)}\ge 0$ and $\beta_i^{(a)}(a-y_i)=0$ due to $y_i=a$, and $\beta_i^{(b)}(y_i-b)=0$.

For $\forall y_i\in D_b$, we have $\forall i:\beta_i^{(a)}=0$ due to the complementary slackness, and

$$y_i=b=\frac{1}{2}(\frac{2A-\sum_i^m\beta_i^{(b)}}{m}+\beta_i^{(b)})$$
$$\Rightarrow \beta_i^{(b)}=2b-\frac{2A-|D_b|\beta_i^{(b)}}{m}=\frac{2(mb-A)}{m-|D_b|}.$$

Since $mb\ge A$ and $|D_b|\le m$, we have $\beta_i^{(b)}\ge 0$ for all $y_i\in D_b$. Therefore, for $\forall i: y_i\in D_b$, we have $\beta_i^{(a)},\beta_i^{(b)}\ge 0$ and $\beta_i^{(a)}(a-y_i)=0$ due to $\beta_i^{(a)}=0$, and $\beta_i^{(b)}(y_i-b)=0$ due to $y_i=b$. For $y_i\in D_{ab}$, we have $\beta_i^{(a)}=\beta_i^{(b)}=0$ due to the complementary slackness.

In conclusion, for $\forall i: y_i\in D$, the partial derivative $\frac{\partial L}{\partial y_i}$ is zero (satisfying **Stationarity**), the sum $\sum_{i=1}^{m} y_i=A$, and $\forall i: a\le y_i\le b$ (satisfying **Primal feasibility**), the constants $\beta_i^{(a)},\beta_i^{(b)}\ge 0$ for all $y_i\in D$ (satisfying **Dual feasibility**), and $\beta_i^{(a)}(a-y_i)=0, \beta_i^{(b)}(y_i-b)=0$ for all $y_i\in D$ (satisfying **Complementary slackness**).

B Error of IPA on SR

We first analyze the error of $\hat{\mu}_t$ under the SR mechanism. In SR, the estimated mean $\hat{\mu}_t$ after the attack is

$$\frac{2}{n+m}\left(\sum_{i=1}^{n_1}\Phi_1(\Psi(x_{i,(1)}))+\sum_{i=1}^{m_1}\Phi_1(\Psi(y_{i,(1)}))\right)=\hat{\mu}_t.$$

Thus we have the expectation of $\hat{\mu}_t$

$$\mathbb{E}(\hat{\mu}_t)=\frac{2}{m+n}\mathbb{E}\left[\sum_{i=1}^{n_1}\Phi_1(\Psi(x_{i,(1)}))+\sum_{i=1}^{m_1}\Phi_1(\Psi(y_{i,(1)}))\right]$$
$$=\frac{2}{m+n}\mathbb{E}\left[\mathbb{E}\left[\sum_{i=1}^{n_1}\Phi_1(\Psi(x_{i,(1)}))+\sum_{i=1}^{m_1}\Phi_1(\Psi(y_{i,(1)}))\,\middle|\, g_1\right]\right]$$
$$=\frac{2\mathbb{E}\left[\sum_{i=1}^{n_1}x_{i,(1)}+\sum_{i=1}^{m_1}y_{i,(1)}\right]}{m+n}=\frac{m+n_e}{m+n}\mu_t+\frac{(S^{(1)}-S_e^{(1)})}{m+n}.$$

Then we can calculate the error as $\mathbb{E}[(\hat{\mu}_t-\mu_t)^2]=Var[\hat{\mu}_t]+(\mathbb{E}(\hat{\mu}_t)-\mu_t)^2$. The bias is known due to $\mathbb{E}(\hat{\mu}_t)=\frac{m+n_e}{m+n}\mu_t+\frac{1}{m+n}(S^{(1)}-X_e)$. Here we study the term $Var[\hat{\mu}_t]$. We denote the $\Phi(\Psi())$ by $M()$. Let $\sum_{i=1}^{n_1}\Phi_1(\Psi(x_{i,(1)}))$ and $\sum_{i=1}^{m_1}\Phi_1(\Psi(y_{i,(1)}))$ be $M(X_{g_1})$ and $M(Y_{g_1})$ respectively. We also denote $\sum_{i=1}^{n_1}x_i$ and $\sum_{i=1}^{m_1}y_i$ as X_{g_1} and Y_{g_1}. Thus, we have

$$Var[\hat{\mu}_t]=\mathbb{E}\left[(\hat{\mu}_t-\mathbb{E}(\hat{\mu}_t))^2\right]$$
$$=\mathbb{E}\left[\left(\frac{2}{n+m}(M(X_{g_1})+M(Y_{g_1}))-\frac{2}{m+n}(X_{g_1}+Y_{g_1}\right)^2\right]$$
$$+\mathbb{E}\left[\left(\frac{2}{m+n}(X_{g_1}+Y_{g_1})-\mathbb{E}(\hat{\mu}_t)\right)^2\right]$$
$$+2\mathbb{E}\left[\left(\frac{2}{n+m}(M(X_{g_1})+M(Y_{g_1}))-\frac{2}{m+n}(X_{g_1}+Y_{g_1})\right)\right.$$
$$\left.\times\left(\frac{2}{m+n}(X_{g_1}+Y_{g_1})-\mathbb{E}(\hat{\mu}_t)\right)\right].$$

The variance contains three terms. For the first term,

$$\mathbb{E}\left[\left(\frac{2}{n+m}(M(X_{g_1})+M(Y_{g_1}))-\frac{2}{m+n}(X_{g_1}+Y_{g_1})\right)^2\right]$$
$$=\frac{2}{(m+n)^2(p-q)^2}\left(m+n-(p-q)^2\left(S^{(2)}+\sum_{i=1}^{m}y_i^2\right)\right).$$

The first equality is based on Lemma 1. Since $\sum_{i=1}^{m}y_i^2 = (n_e+m)(\mu_t^2+\sigma_t^2)-S_e^{(2)}$, the first term equals

$$\frac{2}{(m+n)(p-q)^2}-2\frac{(n_e+m)(\mu_t^2+\sigma_t^2)}{(m+n)^2}-2\frac{S^{(2)}-S_e^{(2)}}{(m+n)^2}.$$

From the standard analysis on sampling process, the second term equals $\frac{n_e+m}{(m+n)^2}(\mu_t^2+\sigma_t^2)+\frac{1}{(m+n)^2}(S^{(2)}-S_e^{(2)})$. Since $\mathbb{E}[M(X_{g_1})]=\mathbb{E}[X_{g_1}]$, $\mathbb{E}[M(Y_{g_1})]=\mathbb{E}[Y_{g_1}]$ and $\mathbb{E}(\hat{\mu}_t)$ is a constant, we have the third term being zero. Therefore, based on the above three terms, we have the error

$$\mathbb{E}[(\hat{\mu}_t-\mu_t)^2]=\left(\frac{n_e-n}{m+n}\mu_t+\frac{(S^{(1)}-S_e^{(1)})}{(m+n)}\right)^2$$
$$+\frac{2}{(m+n)(p-q)^2}-\frac{(n_e+m)(\mu_t^2+\sigma_t^2)}{(m+n)^2}-\frac{S^{(2)}-S_e^{(2)}}{(m+n)^2}$$

Then we study the error of $\hat{\sigma}_t^2$ under the SR mechanism. Denote $\sum_{i=1}^{n_2}\Phi_2(\Psi(x_{i,(2)}^2))$ and $\sum_{i=1}^{m_2}\Phi_2(\Psi(y_{i,(2)}^2))$ by $M(X_{g_2})$ and $M(Y_{g_2})$, and $\sum_{i=1}^{n_2}x_i^2$ and $\sum_{i=1}^{m_2}y_i^2$ by X_{g_2} and Y_{g_2} respectively. The estimated variance (after attack) can be written as $\frac{2}{n+m}(M(X_{g_2})+M(Y_{g_2}))-\hat{\mu}_t^2=\hat{\sigma}_t^2$. Thus we have the expectation of $\hat{\sigma}_t^2$

$$\mathbb{E}(\hat{\sigma}_t^2)=\frac{2}{m+n}\mathbb{E}[M(X_{g_2})+M(Y_{g_2})]-\mathbb{E}[\hat{\mu}_t^2]$$
$$=\frac{m+n_e}{m+n}(\mu_t^2+\sigma_t^2)+\frac{(S^{(2)}-S_e^{(2)})}{m+n}-(Var(\hat{\mu}_t)+\mathbb{E}[\hat{\mu}_t]^2).$$

We can calculate the error $\mathbb{E}[(\hat{\sigma}_t^2-\sigma_t^2)^2]=Var[\hat{\sigma}_t^2]+(\mathbb{E}(\hat{\sigma}_t^2)-\sigma_t^2)^2$. The bias is also known since the expectation $\mathbb{E}[\hat{\sigma}_t^2]$ is known. Next we study the term $Var[\hat{\sigma}_t^2]$

$$Var[\hat{\sigma}_t^2]=Var\left[\frac{2}{m+n}(M(X_{g_2})+M(Y_{g_2}))\right]+Var[\hat{\mu}_t^2]$$

Similar to the analysis of $Var[\hat{\mu}_t]$ which is $Var[\frac{2}{m+n}(M(X_{g_1})+M(Y_{g_1}))]$, we denote the $\sum_{i=1}^{n}x_i^4$ by $S^{(4)}$ and have $Var\left[\frac{2(M(X_{g_2})+M(Y_{g_2}))}{m+n}\right]$ equal $\frac{2}{(m+n)(p-q)^2}-\frac{S^{(4)}+\sum_{i=1}^{m}y_i^4}{(m+n)^2}$. Since each $y_i^4\geq 0$, we have $\sum_{i=1}^{m}y_i^4\geq 0$. For the term $Var[\hat{\mu}_t^2]=\mathbb{E}[\hat{\mu}_t^4]-\mathbb{E}[\hat{\mu}_t^2]^2$, we have $\mathbb{E}[\hat{\mu}_t^2]^2\geq 0$ and $\hat{\mu}_t\leq b$. Thus, it is bounded by $Var[\hat{\mu}_t^2]=\mathbb{E}[\hat{\mu}_t^4]-\mathbb{E}[\hat{\mu}_t^2]^2\leq\mathbb{E}[\hat{\mu}_t^4]\leq 1$.

Given $Var[\hat{\mu}_t^2]$, $Var\left[\frac{2}{m+n}(M(X_{g_2})+M(Y_{g_2}))\right]$ and $\mathbb{E}(\hat{\sigma}_t^2)$, we have the upper bound of error $\mathbb{E}[(\hat{\sigma}_t^2-\sigma_t^2)^2]$.

C Error of OPA on SR

We first analyze the error of $\hat{\mu}_t$ under the SR mechanism. The estimated mean $\hat{\mu}_t$ after the attack is

$$\frac{2}{n+m}\left(\sum_{i=1}^{n_1}\Phi_1(\Psi(x_{i,(1)}))+\sum_{i=1}^{m_1}\Phi_1(\Psi(y_{i,(1)}))\right)=\hat{\mu}_t.$$

Thus the expectation of $\hat{\mu}_t$ is $\mathbb{E}(\hat{\mu}_t)=\frac{m+n_e}{m+n}\mu_t+\frac{1}{m+n}(S^{(1)}-S_e^{(1)})$. The error is calculated as $\mathbb{E}[(\hat{\mu}_t-\mu_t)^2]=Var[\hat{\mu}_t]+(\mathbb{E}(\hat{\mu}_t)-\mu_t)^2$. The bias is known due to $\mathbb{E}(\hat{u}_t)=\frac{m+n_e}{m+n}\mu_t+\frac{1}{m+n}(S^{(1)}-S_e^{(1)})$. Here we study the term $Var[\hat{\mu}_t]$. We denote the $\Phi(\Psi())$ by $M()$, and $\sum_{i=1}^{n_1}\Phi_1(\Psi(x_{i,(1)}))$ and $\sum_{i=1}^{m_1}\Phi_1(\Psi(y_{i,(1)}))$ by $M(X_{g_1})$ and $M(Y_{g_1})$. Let $\sum_{i=1}^{n_1}x_i$ and $\sum_{i=1}^{m_1}y_i$ be X_{g_1} and Y_{g_1}. We have

$$Var[\hat{\mu}_t]=Var\left[\frac{2}{n+m}(M(X_{g_1})+M(Y_{g_1}))\right].$$

Since the adversary directly crafts the output values, the term $M(Y_{g_1})$ is a constant, which can be ignored in the variance. Therefore, the variance

$$Var[\hat{\mu}_t]=\frac{4}{(m+n)^2}\mathbb{E}\left[\left(M(X_{g_1})-X_{g_1}+X_{g_1}-\frac{S^{(1)}}{2}\right)^2\right].$$

We then calculate the expected value,

$$\mathbb{E}\left[\left(M(X_{g_1})-X_{g_1}+X_{g_1}-\frac{S^{(1)}}{2}\right)^2\right]$$
$$=\mathbb{E}\left[(M(X_{g_1})-X_{g_1})^2\right]+\mathbb{E}\left[\left(X_{g_1}-\frac{S^{(1)}}{2}\right)^2\right]$$
$$+2\mathbb{E}\left[(M(X_{g_1})-X_{g_1})\times\left(X_{g_1}-\frac{S^{(1)}}{2})\right)\right].$$

It contains three terms. For the first term,

$$\mathbb{E}\left[(M(X_{g_1})-X_{g_1})^2\right]=\mathbb{E}\left[\mathbb{E}\left[(M(X_{g_1})-X_{g_1})^2\mid g_1\right]\right]$$
$$=\mathbb{E}\left[\frac{1}{(p-q)^2}\left(n_1-(p-q)^2\left(\sum_{i=1}^{n_1}x_i^2\right)\right)\right]=\frac{(\frac{n}{2}-\frac{(p-q)^2}{2}S^{(2)})}{(p-q)^2}.$$

The second equality is based on Lemma 1. From the standard analysis on sampling process, the second term is

$$\mathbb{E}\left[\left(X_{g_1}-\frac{S^{(1)}}{2}\right)^2\right]=\frac{1}{4}\mathbb{E}\left[\left(2X_{g_1}-S^{(1)}\right)^2\right]=\frac{1}{4}S^{(2)}.$$

Since $\mathbb{E}[M(X_{g_1})]=\mathbb{E}[X_{g_1}]$ and $\frac{S^{(1)}}{2}$ is a constant, the third term is zero. Therefore, given the above three terms, we have

the error

$$\mathbb{E}[(\hat{\mu}_t-\mu_t)^2]=\frac{\left(2n-2(p-q)^2S^{(2)}\right)}{(m+n)^2(p-q)^2}+\frac{S^{(2)}}{(m+n)^2}$$
$$+\left(\frac{n_e-n}{m+n}\mu_t+\frac{(S^{(1)}-S_e^{(1)})}{(m+n)}\right)^2.$$

We study the error of $\hat{\sigma}_t^2$ in SR. We denote $\sum_{i=1}^{n_2}\Phi_2(\Psi(x_{i,(2)}^2))$ and $\sum_{i=1}^{m_2}\Phi_2(\Psi(y_{i,(2)}^2))$ by $M(X_{g_2})$ and $M(Y_{g_2})$, and let $\sum_{i=1}^{n_2}x_i^2$ and $\sum_{i=1}^{m_2}y_i^2$ by X_{g_2} and Y_{g_2}. The estimated variance (after the attack) can be written as

$$\frac{2}{n+m}\left(M(X_{g_2})+M(Y_{g_2})\right)-\hat{\mu}_t^2=\hat{\sigma}_t^2.$$

Thus we have the expectation of $\hat{\sigma}_t^2$

$$\mathbb{E}(\hat{\sigma}_t^2)=\frac{2}{m+n}\mathbb{E}\left[M(X_{g_2})+M(Y_{g_2})\right]-\mathbb{E}[\hat{\mu}_t^2]$$
$$=\frac{m+n_e}{m+n}(\mu_t^2+\sigma_t^2)+\frac{(S^{(2)}-S_e^{(2)})}{m+n}-(Var(\hat{\mu}_t)+\mathbb{E}[\hat{\mu}_t]^2).$$

We calculate the error $\mathbb{E}[(\hat{\sigma}_t^2-\sigma_t^2)^2]=Var[\hat{\sigma}_t^2]+(\mathbb{E}(\hat{\sigma}_t^2)-\sigma_t^2)^2$. The bias is known since we know the expectation $\mathbb{E}[\hat{\sigma}_t^2]$. Next we work on the term $Var[\hat{\sigma}_t^2]$

$$Var[\hat{\sigma}_t^2]=Var\left[\frac{2}{m+n}(M(X_{g_2})+M(Y_{g_2}))\right]+Var[\hat{\mu}_t^2].$$

Similar to the analysis of $Var[\hat{\mu}_t]$ which is $Var[\frac{2}{m+n}(M(X_{g_1})+M(Y_{g_1}))]$, we denote $\sum_{i=1}^{n}x_i^4$ as $S^{(4)}$ and have

$$Var\left[\frac{2}{m+n}(M(X_{g_2})+M(Y_{g_2}))\right]=\frac{4Var\left[M(X_{g_2})\right]}{(m+n)^2}$$
$$=\frac{2n-2(p-q)^2S^{(4)}}{(m+n)^2(p-q)^2}+\frac{S^{(4)}}{(m+n)^2}.$$

For the term $Var[\hat{\mu}_t^2]=\mathbb{E}[\hat{\mu}_t^4]-\mathbb{E}[\hat{\mu}_t^2]^2$, we have $\mathbb{E}[\hat{\mu}_t^2]^2\geq 0$ and $\hat{\mu}_t\leq 1$. Thus, it is bounded by $Var[\hat{\mu}_t^2]=\mathbb{E}[\hat{\mu}_t^4]-\mathbb{E}[\hat{\mu}_t^2]^2\leq\mathbb{E}[\hat{\mu}_t^4]\leq 1$ Given $Var[\hat{\mu}_t^2]$, $Var\left[\frac{2}{m+n}(M(X_{g_2})+M(Y_{g_2}))\right]$ and $\mathbb{E}(\hat{\sigma}_t^2)$, we have the upper bound of error $\mathbb{E}[(\hat{\sigma}_t^2-\sigma_t^2)^2]$.

To show and compare the error of IPA and OPA, we replace some terms with intermediate notations shown in Table 5. The comparison results are shown in Table 2.

D Proof of Theorem 2

Proof. First we study OPA and IPA in the SR mechanism. Given the error analysis of SR and PM, we have $\mathrm{Err}_{IPA}(\hat{\mu}_t)-\mathrm{Err}_{OPA}(\hat{\mu}_t)=\frac{2m}{(m+n)^2(p-q)^2}-\frac{(n_e+m)(\mu_t^2+\sigma_t^2)-S_e^{(2)}}{(m+n)^2}$. Since $(n_e+m)(\mu_t^2+\sigma_t^2)-S_e^{(2)}=\sum_{i=1}^{m}y_i^2\leq m$, we have $\mathrm{Err}_{IPA}(\hat{\mu}_t)-\mathrm{Err}_{OPA}(\hat{\mu}_t)\geq\frac{2m-m(p-q)^2}{(m+n)^2(p-q)^2}\geq 0$.

Table 5: Intermedia for Error Analysis.

Intermedia	Values
$\mathcal{P}$	$\left(\frac{n_e-n}{m+n}\mu_t+\frac{(S^{(1)}-S_e^{(1)})}{(m+n)}\right)^2$
$\mathcal{Q}$	$\frac{(n_e+m)(\mu_t^2+\sigma_t^2)}{(m+n)^2}+\frac{S^{(2)}-S_e^{(2)}}{(m+n)^2}$
$S^{(p)}$	$\sum_{i=1}^{n}x_i^p$
η	$\frac{m+n_e}{m+n}(\mu_t^2+\sigma_t^2)-\sigma_t^2$
$\mathcal{T}_{\mathrm{SR}}^{IPA}$	$\left(\eta+\frac{1}{m+n}(S^{(2)}-S_e^{(2)})-Var_{\mathrm{SR}}^{IPA}[\hat{\mu}_t]-\mathbb{E}[\hat{\mu}_t]^2\right)^2$
$\mathcal{T}_{\mathrm{PM}}^{IPA}$	$\left(\eta+\frac{1}{m+n}(S^{(2)}-S_e^{(2)})-Var_{\mathrm{PM}}^{IPA}[\hat{\mu}_t]-\mathbb{E}[\hat{\mu}_t]^2\right)^2$
$\mathcal{T}_{\mathrm{SR}}^{OPA}$	$\left(\eta+\frac{1}{m+n}(S^{(2)}-S_e^{(2)})-Var_{\mathrm{SR}}^{OPA}[\hat{\mu}_t]-\mathbb{E}[\hat{\mu}_t]^2\right)^2$
$\mathcal{T}_{\mathrm{PM}}^{OPA}$	$\left(\eta+\frac{1}{m+n}(S^{(2)}-S_e^{(2)})-Var_{\mathrm{PM}}^{OPA}[\hat{\mu}_t]-\mathbb{E}[\hat{\mu}_t]^2\right)^2$
$Var_{\mathrm{SR}}^{IPA}[\hat{\mu}_t]$	$\frac{2}{(m+n)(p-q)^2}-\mathcal{Q}$
$Var_{\mathrm{PM}}^{IPA}[\hat{\mu}_t]$	$\frac{2(e^{\varepsilon/2}+3)}{3(n+m)(e^{\varepsilon/2}-1)^2}+\frac{2\mathcal{Q}}{(e^{\varepsilon/2}-1)}+\mathcal{Q}$
$Var_{\mathrm{SR}}^{\mathsf{OPA}}[\hat{\mu}_t]$	$\frac{\left(2n-2(p-q)^2S^{(2)}\right)}{(m+n)^2(p-q)^2}+\frac{S^{(2)}}{(m+n)^2}$
$Var_{\mathrm{PM}}^{\mathsf{OPA}}[\hat{\mu}_t]$	$\frac{2n(e^{\varepsilon/2}+3)}{3(m+n)^2(e^{\varepsilon/2}-1)^2}+\frac{(1+e^{\varepsilon/2})S^{(2)}}{(m+n)^2(e^{\varepsilon/2}-1)}$
$\mathbb{E}[\hat{\mu}_t]$	$\frac{m+n_e}{m+n}\mu_t+\frac{1}{m+n}(S^{(1)}-S_e^{(1)})$

For the PM mechanism, since $\sum_{i=1}^{m}y_i^2\geq 0$, we have

$$\mathrm{Err}_{IPA}(\hat{\mu}_t)-\mathrm{Err}_{OPA}(\hat{\mu}_t)=\frac{2m(e^{\varepsilon/2}+3)}{3(m+n)^2(e^{\varepsilon/2})}+$$
$$\frac{2(\sum_{i=1}^{m}y_i^2+S^{(2)})}{(m+n)^2(e^{\varepsilon/2}-1)}+\frac{\sum_{i=1}^{m}y_i^2+S^{(2)}}{(m+n)^2}-\frac{(1+e^{\varepsilon/2})S^{(2)}}{(m+n)^2(e^{\varepsilon/2}-1)}\geq 0$$

Then we compare the error on variance. Since $Var_{\mathrm{SR}}^{\mathsf{OPA}}\leq Var_{\mathrm{SR}}^{\mathsf{IPA}}$, we have $\mathcal{T}_{\mathrm{SR}}^{OPA}\leq\mathcal{T}_{\mathrm{SR}}^{IPA}$. Besides, we have $\frac{2n-2(p-q)^2S^{(4)}}{(m+n)^2(p-q)^2}+\frac{S^{(4)}}{(m+n)^2}\leq\frac{2}{(m+n)(p-q)^2}-\frac{S^{(4)}}{(m+n)^2}$, then it turns out the upper bound of $\mathrm{Err}_{OPA}(\hat{\sigma}_t^2)\leq\mathrm{Err}_{IPA}(\hat{\sigma}_t^2)$ on SR. By the similar calculation, we have the same conclusion on PM mechanism.

□

E Proof of Theorem 3

Proof. According to the attack error, we calculate the derivative of attack error on mean and the upper bound of the attack error on variance, and have all derivatives negative for all $\varepsilon>0$. In other words, the attack error on mean and the upper bound of attack error on variance decrease as ε grows. □