Budget Inference Attacks and Countermeasures in Locally Differentially Private Data Collection

BERKAY KEMAL BALIOGLU, Computer Engineering, Koç University, Istanbul, Turkey
EMRE GURSOY, Computer Engineering, Koç University, Istanbul, Turkey

Local differential privacy (LDP) has recently become a popular notion for privacy-preserving data collection from user devices. It has been applied in numerous contexts related to the Internet of Things (IoT) and cyber-physical systems to enable privacy-preserving edge data analytics. The strength of privacy protection in LDP deployments depends on the privacy budget ε, and there are several scenarios in which it is desirable for the value of ε to remain hidden from untrusted third parties, or the inference of ε by an untrusted third party may constitute a privacy leakage. In this article, we propose a new class of attacks called budget inference attacks (BIAs), which enable an adversary to infer the ε budget value from the outputs of an LDP protocol. We develop BIAs for two types of adversaries: informed adversaries who have knowledge of the statistical data distribution, and uninformed adversaries who do not. We apply our BIAs on five popular LDP protocols and experimentally evaluate them using multiple datasets, varying ε budgets, population sizes, and attack settings and parameters. Results show that our BIAs are highly effective, as they enable the adversary to infer the ε value with low errors. We also propose three potential countermeasures against our BIAs. Analyses show that while our countermeasures can be effective in reducing BIA accuracy, they also increase utility loss; therefore, the tradeoff between BIA accuracy and utility loss needs to be carefully considered.

CCS Concepts: • **Security and privacy → Privacy-preserving protocols**; • **Computer systems organization → Distributed architectures;**

Additional Key Words and Phrases: Privacy, local differential privacy, attacks, privacy budget, distributed data processing, internet of things

ACM Reference Format:
Berkay Kemal Balioglu and Emre Gursoy. 2026. Budget Inference Attacks and Countermeasures in Locally Differentially Private Data Collection. *ACM Trans. Internet Technol.* 26, 1, Article 7 (January 2026), 32 pages. https://doi.org/10.1145/3759250

1 Introduction

Over the past few years, **local differential privacy** (**LDP**) has rapidly become a popular standard for ensuring privacy in the collection of user data [10, 59, 62]. By enabling data to be perturbed directly on the user's device, whether it be a smartphone, smart meter, or IoT device, LDP ensures that sensitive information is protected before it leaves the user's possession. LDP has seen widespread adoption in numerous fields related to the **Internet of Things** (**IoT**) and cyber-physical systems,

This study was supported by the Scientific and Technological Research Council of Türkiye (TUBITAK) under Grant Number 123E179. The authors thank TUBITAK for their support.
Authors' Contact Information: Berkay Kemal Balioglu, Computer Engineering, Koç University, Istanbul, Turkey; e-mail: bbalioglu23@ku.edu.tr; Emre Gursoy, Computer Engineering, Koç University, Istanbul, Turkey; e-mail: emregursoy@ku.edu.tr.



This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

© 2026 Copyright held by the owner/author(s).
ACM 1533-5399/2026/01-ART7
https://doi.org/10.1145/3759250

such as geolocation data [12, 24, 52], indoor positioning [30, 38], health data and wearables [31, 36, 57], Internet of Vehicles [63], cybersecurity [23], smart meters and smart grids [19, 41], as well as distributed deep learning [2, 49, 51]. It has also been deployed in several popular industry products such as Chrome, Apple iOS, and Microsoft Windows [1, 11, 14].

The strength of privacy protection in LDP depends on the privacy budget parameter ε. While ε is not necessarily a secret parameter in LDP, there are cases in which the value of ε may be hidden from third parties, or the inference of ε by an untrusted third party may constitute a privacy leakage. For example, a company or developer may implement LDP in their product but may not want the ε budgets to be known by untrusted third parties. (This was actually the case in the early days of Apple's implementation of LDP [45, 48].) Another example is **personalized LDP** (**PLDP**), in which users select ε in a personalized fashion based on their preferences and desired level of privacy protection [40, 44, 46]. The adversary's inference that one user applies $\varepsilon = 0.5$ but another user applies $\varepsilon = 2$ may implicitly disclose users' privacy preferences and/or the sensitivity of their values. Furthermore, knowledge of ε can also benefit the adversary in executing some other attacks (e.g., poisoning attacks) more effectively. Further examples and details regarding why budget inference may constitute a threat are given in Section 2.3.

In this article, we propose a new class of attacks called **Budget Inference Attacks** (**BIAs**), which enable an adversary to infer the ε budget values from LDP protocol outputs. We develop BIAs for two types of adversaries: informed adversaries who have knowledge of the statistical data distribution f in the user population, and uninformed adversaries who do not have knowledge of f. We formulate our BIAs for informed adversaries in a generic (protocol-independent) way. In these BIAs, the adversary first uses the observed protocol outputs to estimate the observed distribution $\tilde{\mathrm{f}}$ under LDP and calculates the distance between f and $\tilde{\mathrm{f}}$ as the error amount. Then, the adversary searches for the ε value that would yield the error amount closest to the observed error. Within each step of the search, the adversary internally simulates an LDP data collection (using a synthetic user population, without the involvement of real users) with the current ε and notes the error. If the error is too high compared to the observed error, then the real ε value should be larger than the adversary's current guess; therefore, larger ε values are searched in the next iterations. Otherwise, smaller ε values are searched.

Considering that the informed BIA strategy is not applicable to uninformed adversaries, we propose custom-tailored uninformed BIAs for various popular LDP protocols. These BIAs exploit the protocols' inner principles and parameters. For example, in the bitvector-based RAPPOR and OUE protocols, the probability of bit keeping and bit flipping is known. Lower ε increases the bit flipping probability, and will therefore, result in a higher number of one bits. By counting the total number of zero bits and one bits in the users' protocol outputs, and comparing these counts to the expected number of one bits under different ε, the adversary can predict the true ε. On the other hand, protocols such as OLH and SW have internal parameters which are closely tied to the value of ε, e.g., hash parameter g in OLH and bound parameter b in SW. An adversary can predict the value of ε by estimating the values of g and b from users' reported protocol outputs and then leveraging these estimates to infer the value of ε.

We experimentally evaluate our informed and uninformed BIAs using five popular LDP protocols (GRR, OLH, RAPPOR, OUE, SW), multiple datasets, and varying ε budgets. Results show that our BIAs are highly effective, e.g., we observe that both informed and uninformed BIAs achieve **Mean Absolute Error** (**MAE**) strictly lower than 0.03 in more than 73% of the experiments, and MAE strictly lower than 0.01 in more than 50% of the experiments. In the majority of the protocols, informed BIAs are similar or superior to uninformed BIAs in terms of MAE, which shows that the availability of f contributes positively to executing BIAs with higher effectiveness. Through extensive experiments, we also evaluate the impacts of varying protocols, real ε budgets, and attack

parameters on BIA effectiveness. Furthermore, our analyses with varying population sizes show that in several protocols, MAEs remain low despite the existence of only a few users using the same ε (e.g., 10–50 users). This result indicates that our attacks are also effective in PLDP scenarios or custom IoT data collection architectures where only a few users may be participating or only a few users may be using the same ε.

We then study potential countermeasures against our BIAs. We propose and evaluate three countermeasures: noisy knowledge of f, usage of suboptimal protocol parameters, and usage of suboptimal perturbation probabilities. Our analyses show that while these countermeasures can be effective in reducing the accuracy of BIAs, they come at the cost of higher noise and reduced utility in the aggregate statistics learned under LDP. These are likely to have negative impacts on the use of collected data for downstream legitimate tasks. Hence, the tradeoff between BIA accuracy reduction and utility loss should be managed carefully.

Contributions. In short, the contributions of our article can be summarized as follows:

- We propose a novel class of attacks called BIAs, which enable an adversary to infer the ε budget values from LDP protocol outputs.
- We propose BIAs for two types of adversaries (informed and uninformed) based on their knowledge of f.
- We experimentally evaluate our informed and uninformed BIAs using five popular LDP protocols, five datasets, and varying ε budgets. Results show that our BIAs are highly effective.
- We propose and evaluate three countermeasures against our BIAs. Results show that our countermeasures can be effective in reducing the accuracy of BIAs, but they usually come at the cost of reduced data utility.

Article organization. The rest of this article is organized as follows. In Section 2, we present background and preliminaries regarding LDP, LDP protocols, and the goal of BIAs. In Section 3, we present our BIAs for informed adversaries. In Section 4, we present our BIAs for uninformed adversaries. We report our experiment results and discussion in Section 5. We propose and evaluate our countermeasures against BIAs in Section 6. We present related work in Section 7 and conclude the article in Section 8.

2 Preliminaries

2.1 Local Differential Privacy (LDP)

LDP has recently emerged as a popular technique for privacy-preserving data collection [10, 54, 59]. In LDP, each user perturbs their data before sending it to the server (data collector). Since perturbation occurs on the user side, LDP can be used in scenarios where users do not trust the data collector. After receiving and aggregating perturbed data from many users, the data collector can apply estimation algorithms to recover statistics pertaining to the general user population, but cannot directly infer the true value of any particular user.

Let $\mathcal{P}$ denote the user population and let $\mathcal{U}$ denote the domain of users' true values. For user $u \in \mathcal{P}$, we denote this user's true value by v_u, such that $v_u \in \mathcal{U}$. In order to satisfy LDP, each user perturbs his/her v_u using a randomized algorithm ψ and then sends the perturbed outcome to the server. For the algorithm ψ to satisfy LDP, the following definition must hold.

Definition 1 (ε-LDP). A randomized mechanism ψ satisfies ε-LDP, if and only if for any two values v_1, v_2 in $\mathcal{U}$:

$$\forall y \in Range(\psi) \; : \; \frac{\Pr[\psi(v_1) = y]}{\Pr[\psi(v_2) = y]} \leq e^{\varepsilon}, \tag{1}$$

where $Range(\psi)$ denotes the set of all possible outputs of ψ.

ε-LDP ensures that, having observed the output y, the data collector is not able to distinguish whether the original value was v_1 or v_2 with probability odds ratio higher than e^{ε}. Here, $\varepsilon > 0$ is the key privacy parameter (also called the *privacy budget*) that controls the strength of privacy protection. Lower ε yields stronger privacy, whereas higher ε yields weaker privacy.

2.2 LDP Protocols

Various LDP protocols were designed in the literature to minimize utility loss and/or communication cost under different conditions [6, 10, 14, 54]. These protocols typically serve as building blocks for downstream data analysis tasks and more complex end applications. In this article, we chose five popular and commonly used LDP protocols to demonstrate our attacks: GRR, RAPPOR, OUE, OLH, and SW. Next, we briefly describe these protocols.

Generalized Randomized Response (GRR). Randomized response is a survey collection technique first introduced by Warner in [56]. GRR is a generalization of this technique tailored for LDP to support multi-valued domains (i.e., $|\mathcal{U}| \geq 3$) and arbitrary ε values. Given the user's true value v_u, GRR outputs a perturbed value $y_u \in \mathcal{U}$ with probability:

$$\Pr[\psi(v_u) = y_u] = \begin{cases} p = \frac{e^{\varepsilon}}{e^{\varepsilon}+|\mathcal{U}|-1}, & \text{if } y_u = v_u \\ q = \frac{1}{e^{\varepsilon}+|\mathcal{U}|-1}, & \text{if } y_u \neq v_u \end{cases}. \tag{2}$$

This satisfies ε-LDP because $\frac{p}{q} = e^{\varepsilon}$. The user sends y_u to the server.

On the server side, upon receiving perturbed values from all users, to perform estimation for some value $v \in \mathcal{U}$, the server first computes $\hat{C}(v)$: number of users who reported v as their perturbed output. Then, the estimate $\tilde{C}(v)$ is computed as

$$\tilde{C}(v) = \frac{\hat{C}(v) - |\mathcal{P}| \cdot q}{p - q}. \tag{3}$$

Randomized Aggregatable Privacy-Preserving Ordinal Response (RAPPOR). RAPPOR protocol was introduced by Google in [14]. In RAPPOR, a user's value is encoded into a bitvector and the bitvector is perturbed in a randomized fashion to satisfy LDP. Although the original RAPPOR protocol uses Bloom filters to encode users' values, here we give a simpler version of RAPPOR with unary encoding that is commonly used in the literature [22, 23, 54].

User u initializes bitvector B_u with length $|B_u| = |\mathcal{U}|$. All bits in B_u are initialized as 0, except for the bit corresponding to v_u: $B_u[v_u] = 1$. Then, the perturbation mechanism of RAPPOR takes B_u as input and produces a perturbed bitvector B'_u by iterating through each index $i \in [1, |B_u|]$, and either keeping the bit $B_u[i]$ of flipping it with probabilities:

$$\forall_{i \in [1,|B_u|]} : \Pr[B'_u[i] = 1] = \begin{cases} \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}+1}, & \text{if } B_u[i] = 1 \\ \frac{1}{e^{\varepsilon/2}+1}, & \text{if } B_u[i] = 0 \end{cases}. \tag{4}$$

The user sends B'_u to the server. The server receives perturbed bitvectors from many users $u \in \mathcal{P}$. Afterward, to perform estimation for index i, the server first finds $\bar{C}[i]$ as

$$\bar{C}[i] = \sum_{u \in \mathcal{P}} B'_u[i]. \tag{5}$$

Then, the estimate $\tilde{C}(i)$ is computed as

$$\tilde{C}(i) = \frac{\bar{C}[i] + |\mathcal{P}| \cdot (\alpha - 1)}{2\alpha - 1}, \tag{6}$$

where α is the bit keeping probability: $\alpha = \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}+1}$.

Optimized Unary Encoding (OUE). The bitvector encoding phase of the OUE protocol is the same as RAPPOR explained above, but the probabilities of keeping or flipping the bits are different. The probabilities in OUE were derived mathematically in [54] to minimize the variance of the estimations performed on the server side, thereby improving the utility of the protocol. That is, after setting up the bitvector B_u in the same way as RAPPOR, the perturbed bitvector B'_u is determined according to the following probabilities:

$$\forall_{i\in[1,|B_u|]} \; : \; \Pr[B'_u[i]=1] = \begin{cases} \frac{1}{2} & \text{if } B_u[i]=1 \\ \frac{1}{e^{\varepsilon}+1} & \text{if } B_u[i]=0 \end{cases}. \tag{7}$$

The user sends B'_u to the server. After receiving perturbed bitvectors from many users $u \in \mathcal{P}$, to perform estimation for index i, $\tilde{C}[i]$ is computed in the same way as Equation (5). Then, the estimate $\tilde{C}(i)$ is found as

$$\tilde{C}(i) = \frac{2\cdot\left((e^{\varepsilon}+1)\cdot\bar{C}[i]-|\mathcal{P}|\right)}{e^{\varepsilon}-1}. \tag{8}$$

Optimized Local Hashing (OLH). Both RAPPOR and OUE use bitvectors of length $|\mathcal{U}|$ when sending data from the user to the server. They can have relatively high communication cost when $\mathcal{U}$ is large. Instead, OLH uses hash functions to reduce the domain size. To address the issue of recurring hash collisions, OLH utilizes a family of hash functions $\mathcal{H}$ such that each user chooses a different hash function from $\mathcal{H}$.

Let $\mathcal{H}$ be a hash function family such that each $H \in \mathcal{H}$ maps a value from $\mathcal{U}$ into an integer between 1 and g, i.e., $H : \mathcal{U} \to [1,g]$. Here, g is a parameter of the OLH protocol, whose value is $g = \text{INT}(e^{\varepsilon}+1)$ as established in [54, 55].[1] User u with true value v_u chooses a hash function from $\mathcal{H}$ uniformly at random $H_u \leftarrow_{\$} \mathcal{H}$ and computes the integer $x_u \leftarrow H_u(v_u)$. Then, the perturbation step of OLH takes x_u as input and produces a perturbed integer $x'_u \in [1,g]$ with the following probabilities:

$$\forall_{i\in[1,g]} \; : \; \Pr[x'_u=i] = \begin{cases} \frac{e^{\varepsilon}}{e^{\varepsilon}+g-1} & \text{if } x_u=i \\ \frac{1}{e^{\varepsilon}+g-1} & \text{if } x_u\neq i \end{cases}. \tag{9}$$

The user sends the tuple $\langle H_u, x'_u\rangle$ to the server. After receiving tuples from many users $u \in \mathcal{P}$, to perform estimation for value v, the server first computes $Sup(v)$ = the total number of tuples for which the condition $x'_u = H_u(v)$ holds. Then, the estimate $\tilde{C}(v)$ is computed as

$$\tilde{C}(v) = \frac{(e^{\varepsilon}+g-1)\cdot(g\cdot Sup(v)-|\mathcal{P}|)}{(e^{\varepsilon}-1)\cdot(g-1)}. \tag{10}$$

Square Wave (SW). In the SW protocol proposed in [35], the domain $\mathcal{U}$ is assumed to be ordinal, and the user's true value v_u is perturbed in such a way that values closer to v_u have higher probability of being supported rather than values farther away from v_u. On the user side, v_u is first normalized into the range [0,1]. Then, the perturbation algorithm ψ takes the normalized v_u as input and produces the perturbed output v'_u such that

$$\Pr\left[\psi(v_u)=v'_u\right] = \begin{cases} \frac{e^{\varepsilon}}{2be^{\varepsilon}+1}, & \text{if } |v_u-v'_u|\le b \\ \frac{1}{2be^{\varepsilon}+1}, & \text{otherwise.} \end{cases} \tag{11}$$

Here, b is a parameter of the SW protocol, whose value was derived and established in [35]:

$$b = \frac{\varepsilon e^{\varepsilon}-e^{\varepsilon}+1}{2e^{\varepsilon}(e^{\varepsilon}-1-\varepsilon)}. \tag{12}$$

[1] We use $\text{INT}(\cdot)$ to denote rounding to an integer.

The users send their v'_u to the server. After receiving v'_u values from many users $u \in \mathcal{P}$, the server needs to perform estimation. In SW, estimation is performed using a **maximum likelihood estimator** (**MLE**) which takes as input v'_u values from all users and finds estimates $\tilde{C}(v)$ for all $v \in \mathcal{U}$ such that the log-likelihood of obtaining the observed v'_u values is maximized. For brevity we do not include the details of the MLE algorithm in this article, but instead we refer the readers to Section 5.5 of [35].

2.3 Threat Model and Adversary Types

The strength of privacy protection in LDP depends on the value of ε. While ε is not necessarily a secret parameter, there exist several practical scenarios in which the value of ε may be intentionally hidden from external parties, or the inference of ε by untrusted parties may constitute a privacy breach. Below, we outline example scenarios that illustrate budget inference as a threat:

- Companies deploying LDP-based mechanisms may prefer not to disclose their chosen ε budgets to untrusted third parties. For example, in the early stages of Apple's LDP deployment, the ε budgets for the various types of data collected from iOS devices were not made public. Researchers performed static and dynamic code analysis of the macOS Sierra implementation to reverse-engineer and infer Apple's choices regarding privacy budgets [45, 48].
- PLDP is an emerging paradigm in which users select individualized ε budgets based on their privacy peferences and desired levels of privacy protection [40, 44, 46]. Users are inclined to select high ε for information they deem insensitive, but low ε for information they deem sensitive. In such cases, an adversary who learns the ε value dedicated by a user to certain information may infer the user's privacy preferences or how sensitive the user's information is.
- Third, an emerging trend is the *shuffler* architecture, in which a shuffler sits between the users and the data collector. It receives perturbed data from users and randomly shuffles them to achieve anonymity and privacy amplification [4, 13, 16]. An honest-but-curious shuffler may peek into users' perturbed data to infer the ε budgets, even though it does not need ε to perform shuffling.
- The notion of w-event privacy is recently being applied to IoT and edge analytics for the collection of streaming data, e.g., data from sensors or wearables [15, 42]. In w-event privacy, the ε budget can be divided non-uniformly among a window of w events due to the amount of change in the stream, sensitivity of the event, and utility requirements. An adversary's inference of the different ε used in different events could, therefore, leak information about the amount of change, data sensitivity, or the utility requirement.
- Finally, it is possible that the inference of ε can increase the performance of some practical attacks such as membership inference attacks, attribute inference attacks, or poisoning attacks. We experimentally demonstrate this for a popular poisoning attack, MGA [7], in Section 5.7.

In this article, we propose a class of attacks called BIAs, which enable an adversary to infer the ε budget values from LDP protocol outputs. We propose BIAs for two types of adversaries: informed and uninformed. Both types of adversaries are able to see the perturbed data sent from the users to the server. We denote these observations by $\mathcal{O}$, e.g., in case of GRR the adversary observes $\mathcal{O} = \{y_1, y_2, \ldots, y_{|\mathcal{P}|}\}$; in case of RAPPOR the adversary observes $\mathcal{O} = \{B'_1, B'_2, \ldots, B'_{|\mathcal{P}|}\}$; in case of OLH the adversary observes $\mathcal{O} = \{\langle H_1, x'_1\rangle, \langle H_2, x'_2\rangle, \ldots, \langle H_{|\mathcal{P}|}, x'_{|\mathcal{P}|}\rangle\}$, and so forth. Armed with $\mathcal{O}$, the goal of the adversary is to correctly infer the value of the privacy budget ε. The key difference between the informed and uninformed adversaries is that the informed adversary also knows the frequency distribution of the items in the domain: $\mathrm{f} = \{f_v\}_{v \in \mathcal{U}}$. We present our BIAs for informed adversaries in Section 3 and uninformed adversaries in Section 4.

ALGORITHM 1: BIA for informed adversary

Input : $\mathrm{f}, \mathcal{O}, \varepsilon_{\min}, \varepsilon_{\max}, \tau$
Output: Predicted ε

$\tilde{\mathrm{f}} \leftarrow$ AGGREGATE($\mathcal{O}$)
$target \leftarrow$ CALC_DISTANCE($\mathrm{f}, \tilde{\mathrm{f}}$)
$\mathcal{P}_{syn} \leftarrow$ Generate synthetic user population with true value frequency distribution f
Initialize $\varepsilon_{\text{low}} \leftarrow \varepsilon_{\min}$ and $\varepsilon_{\text{high}} \leftarrow \varepsilon_{\max}$
while $\varepsilon_{high} - \varepsilon_{low} > \tau$ **do**
  $\varepsilon_{\text{mid}} \leftarrow \frac{\varepsilon_{\text{high}} + \varepsilon_{\text{low}}}{2}$
  $\mathcal{O}_{syn} \leftarrow$ Simulate LDP data collection with $\mathcal{P}_{syn}$ and budget ε_{mid}
  $\mathrm{f}_{syn} \leftarrow$ AGGREGATE($\mathcal{O}_{syn}$)
  $current \leftarrow$ CALC_DISTANCE($\mathrm{f}, \mathrm{f}_{syn}$)
  if $current < target$ **then**
    $\varepsilon_{\text{high}} \leftarrow \varepsilon_{\text{mid}}$
  else
    $\varepsilon_{\text{low}} \leftarrow \varepsilon_{\text{mid}}$
return ε_{low}

3 Budget Inference Attacks for Informed Adversaries

In this section, we describe our BIAs for informed adversaries. We formulate our BIA algorithm in a generic manner (independent of the underlying LDP protocol), which allows our attack formulation to be applicable to an arbitrary LDP protocol. We present the generic attack formulation in Section 3.1, details regarding the CALC_DISTANCE function used in the attack formulation in Section 3.2, and the theoretical analysis of our BIAs in Section 3.3.

3.1 Generic Attack Formulation

Consider an informed adversary who has observed f and $\mathcal{O}$, and the adversary aims to infer ε. In order to bound the search space for ε, a lower limit $\varepsilon_{\min}$ and an upper limit $\varepsilon_{\max}$ are used so that the adversary will search between $[\varepsilon_{\min}, \varepsilon_{\max}]$ to find ε. Here, note that the bounds can be made arbitrarily large to cover all reasonable ε values that can be used in practice, e.g., $\varepsilon_{\min} = 10^{-5}$, $\varepsilon_{\max} = 20$. The time it takes for our attack to complete is logarithmically dependent on the size of the search space. In addition, a stopping condition parameter τ is used to stop the search so that the search will not continue indefinitely. In our experiments, we use $\tau = 0.001$ by default since we believe that inferring ε with three digits after the decimal point provides more than sufficient precision in many practical, real-world scenarios.

The algorithmic description of our BIA is given in Algorithm 1. Having observed $\mathcal{O}$, the adversary first computes the observed frequencies of all items in the domain: $\tilde{\mathrm{f}} = \{\tilde{f}_v\}_{v \in \mathcal{U}}$ using the AGGREGATE function. Here, $\tilde{f}_v$ can be computed using the aggregation and estimation procedures of each protocol, i.e., Equation (3) for GRR, Equation (6) for RAPPOR, Equation (10) for OLH, and so on. Then:

$$\tilde{f}_v = \frac{\tilde{C}(v)}{\sum_{x \in \mathcal{U}} \tilde{C}(x)}. \tag{13}$$

Afterward, the adversary uses the CALC_DISTANCE function to calculate the distance between f and $\tilde{\mathrm{f}}$. We use multiple methods to calculate the distance between f and $\tilde{\mathrm{f}}$; more details are given in Section 3.2. The distance between f and $\tilde{\mathrm{f}}$ becomes the adversary's *target*. In the remainder of

the attack, the adversary will search for the value of ε which would yield the data collection with distance closest to *target*. To do so, the adversary first generates a synthetic user population $\mathcal{P}_{syn}$ whose true value frequency distribution is equal to f, e.g., by assigning true value equal to v to $f_v \times |\mathcal{P}_{syn}|$ users for all $v \in \mathcal{U}$. Next, the main while loop of the BIA executes (lines 5–13). In each iteration, the current budget ε_{mid} is used to simulate LDP data collection with the synthetic population $\mathcal{P}_{syn}$. Note that this is a *simulation* performed on the adversary's side, which does not involve any real users. Hence, there is no privacy impact on the real users. The adversary internally simulates the execution of an LDP protocol with $\mathcal{P}_{syn}$ and notes the observed results, i.e., simulated results $\mathcal{O}_{syn}$ reported by the simulated user population. $\mathcal{O}_{syn}$ is aggregated by the adversary to obtain f_{syn} (line 8) and the *current* distance between f_{syn} and f is found (line 9).

Here, *current* symbolizes the distance that would have been observed if data was collected from the user population with budget ε_{mid}. If the true ε is close to ε_{mid}, then we expect *target* and *current* to be similar. If ε_{mid} is an overestimation of ε, then we expect *current* to be smaller than *target*, since higher budgets yield lower error (lower noise) in LDP. Thus, lower budgets need to be searched by our BIA in the next iterations. This is achieved by setting $\varepsilon_{\text{high}} \leftarrow \varepsilon_{\text{mid}}$ (line 11), since ε_{mid} will be found as $\frac{\varepsilon_{\text{high}}+\varepsilon_{\text{low}}}{2}$ in the next iteration, therefore reducing the value of $\varepsilon_{\text{high}}$ causes ε_{mid} to be reduced in the next iteration. On the other hand, if *current* $>$ *target*, then the current ε_{mid} is an underestimation. Thus, higher budgets need to be searched by our BIA in the next iterations, which is achieved by setting $\varepsilon_{\text{low}} \leftarrow \varepsilon_{\text{mid}}$ (line 13). This way of updating $\varepsilon_{\text{high}}$ and ε_{low} and iteratively guiding the search is similar to the concept of *binary search*, a well-known search algorithm with complexity logarithmic in the size of the initial search space.

3.2 Implementation of Calc_Distance

The functionality of Calc_Distance is to measure the distance between two frequency distributions, say p and q. We utilize three well-known distance metrics for this purpose: ℓ_1 distance, ℓ_2 distance, and KL-divergence. We perform an experimental comparison between them in Section 5.

ℓ_1 *distance.* Let $\text{p} = [p_1, p_2, \dots, p_n]$ and $\text{q} = [q_1, q_2, \dots, q_n]$ be two frequency distributions. The ℓ_1 distance between p and q can be measured as

$$\ell_1_dist(\text{p}, \text{q}) = \sum_{i=1}^{n} |p_i - q_i|. \tag{14}$$

ℓ_2 *distance.* For p and q defined same way as above, the ℓ_2 distance between them can be measured as

$$\ell_2_dist(\text{p}, \text{q}) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}. \tag{15}$$

***Kullback–Leibler (KL)** divergence* is a popular method to quantify the divergence or dissimilarity between one distribution p and another distribution q. It is formally defined as

$$KL\text{-}divergence(\text{p}, \text{q}) = \sum_{i=1}^{n} p_i \times \log\left(\frac{p_i}{q_i}\right). \tag{16}$$

3.3 Theoretical Analysis

Next, we theoretically analyze our BIA in terms of the expected error in the predicted budget. Throughout the derivations, let $\varepsilon_{\text{true}}$ denote the true budget (ground truth) and $\hat{\varepsilon}$ denote the budget predicted by our BIA. We assume that ℓ_1 distance is used in Calc_Distance, as it is the default choice of distance metric in the rest of the article, and it is experimentally shown (in Section 5) to slightly outperform the other metrics.

Analysis for GRR. We observe from Algorithm 1 that the BIA essentially performs a binary search within the interval $[\varepsilon_{\min}, \varepsilon_{\max}]$, which terminates when the interval becomes smaller than τ. The key source of error throughout the search is the randomness in the ℓ_1 distances computed between real and simulated data caused by LDP. This noise has standard deviation $\sigma = O(\sqrt{\frac{|\mathcal{U}|}{|\mathcal{P}|}})$ since the adversary has observed $|\mathcal{P}|$ samples.

We model the binary search process over $m = \lceil \log_2(R_0/\tau) \rceil$ iterations, where $R_0 = |\varepsilon_{\max} - \varepsilon_{\min}|$ is the initial search interval, and in general, R_j denotes the interval in the j'th iteration. At the j'th iteration, the adversary compares f_{syn} and f (lines 7–9) and changes the search interval accordingly. However, due to LDP randomness, each such comparison has a probability ρ_j of being incorrect. This causes a shift of up to $\frac{1}{2}R_{j-1}$ from the true budget. Combining the fact that GRR's estimation is unbiased and the central limit theorem, we can approximate ρ_j using a Gaussian distribution:

$$\rho_j \approx \Phi\left(-\frac{\ell_1_dist(\mathrm{f}, \tilde{\mathrm{f}}) - \ell_1_dist(\mathrm{f}, \tilde{\mathrm{f}}_{syn})}{\sqrt{2}\sigma}\right), \tag{17}$$

where $\Phi(\cdot)$ is the standard normal CDF. Then, the expected error in the final result becomes:

$$\mathbb{E}[|\hat{\varepsilon} - \varepsilon_{\text{true}}|] \approx \frac{\tau}{4} + \sum_{j=1}^{m} \rho_j \cdot \frac{R_{j-1}}{2}, \tag{18}$$

where $R_{j-1} = \frac{R_0}{2^{j-1}}$ is the search interval width at iteration j. The $\frac{\tau}{4}$ term accounts for the expected rounding error due to returning the midpoint at termination.

Analysis for RAPPOR. For the remaining protocols, the same binary search principle in Algorithm 1 applies, but the standard deviation σ calculation needs to be different. In the case of RAPPOR, the unbiased estimator $\tilde{C}(i)$ has variance:

$$\mathrm{Var}\left[\frac{\tilde{C}(i)}{|\mathcal{P}|}\right] = \frac{\alpha(1-\alpha)}{|\mathcal{P}|\,(2\alpha-1)^2} \quad \text{where} \quad \alpha = \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}+1}. \tag{19}$$

Since ℓ_1 distance sums over all $|\mathcal{U}|$ values, the standard deviation becomes:

$$\sigma = \sqrt{\frac{|\mathcal{U}|\,\alpha(1-\alpha)}{|\mathcal{P}|\,(2\alpha-1)^2}}. \tag{20}$$

The remainder of the analysis is the same as GRR, since it follows from how Algorithm 1 behaves. Hence, Equations (17) and (18) apply. We note that since $\sigma \propto 1/\sqrt{|\mathcal{P}|}$, larger $|\mathcal{P}|$ reduces σ, lowers each ρ_j, and thus decreases the expected error in the predicted budgets.

Analysis for OUE. In the case of OUE, the unbiased estimator has variance:

$$\mathrm{Var}\left[\frac{\tilde{C}(i)}{|\mathcal{P}|}\right] = \frac{(e^{\varepsilon}+1)^2}{|\mathcal{P}|\,(e^{\varepsilon}-1)^2}. \tag{21}$$

Summing over all $|\mathcal{U}|$ values, the standard deviation becomes:

$$\sigma = \sqrt{\frac{|\mathcal{U}|\,(e^{\varepsilon}+1)^2}{|\mathcal{P}|\,(e^{\varepsilon}-1)^2}}. \tag{22}$$

The remainder of the analysis is the same as RAPPOR, i.e., Equations (17) and (18) apply. Similar to RAPPOR, since $\sigma \propto 1/\sqrt{|\mathcal{P}|}$, increasing the population size $|\mathcal{P}|$ reduces σ, decreases each ρ_j, and thus lowers the expected error in the predicted privacy parameter.

Analysis for OLH. In the case of OLH, the unbiased estimator has variance:

$$\mathrm{Var}\left[\frac{\tilde{C}(i)}{|\mathcal{P}|}\right] \leq \frac{(e^{\varepsilon}+g-1)^2\, g^2}{4\,(e^{\varepsilon}-1)^2\,(g-1)^2\,|\mathcal{P}|}. \tag{23}$$

Summing over all $|\mathcal{U}|$ values, the standard deviation becomes:

$$\sigma = O\left(\frac{(e^{\varepsilon}+g-1)\,g}{2\,(e^{\varepsilon}-1)\,(g-1)} \times \sqrt{\frac{|\mathcal{U}|}{|\mathcal{P}|}}\right). \tag{24}$$

The remainder of the analysis is the same as the previous protocols.

Analysis for SW. Since SW uses MLE in its estimation process, a closed-form formula for its variance is not available [35]. Following the previous protocols' derivations and denoting the protocol-specific variance constant of SW by c_{SW}, we write the standard deviation as

$$\sigma = O\left(\sqrt{\frac{c_{\mathrm{SW}}\,|\mathcal{U}|}{|\mathcal{P}|}}\right). \tag{25}$$

The remainder of the analysis is the same as the previous protocols. Notably, SW also contains the $O(\sqrt{\frac{|\mathcal{U}|}{|\mathcal{P}|}})$ factor in σ; therefore, we expect $|\mathcal{U}|$ and $|\mathcal{P}|$ to affect SW similar to how they affect the remaining protocols.

Numerical Examples. In Appendix A, we numerically exemplify our theoretical analyses for two protocols: GRR and RAPPOR.

4 Budget Inference Attacks for Uninformed Adversaries

In this section, we describe our BIAs for uninformed adversaries. Considering that uninformed adversaries do not know f, they cannot use Algorithm 1 because: (i) they cannot calculate the distance between f and $\tilde{\mathrm{f}}$ as well as f and f_{syn}, and (ii) they cannot generate $\mathcal{P}_{syn}$ with frequency distribution equal to f. Thus, attacks explained in the previous section cannot be used by uninformed adversaries. As a result, for uninformed adversaries, we propose protocol-specific, customized attack strategies by taking into account the technical principles of different LDP protocols.

4.1 BIAs for RAPPOR and OUE

We first consider the two bitvector-based LDP protocols: RAPPOR and OUE. For these two protocols, we make the following observation. In both RAPPOR and OUE, before perturbation, the user's original bitvector B_u contains 1 one bit and $|\mathcal{U}|-1$ zero bits. Per Equations (4) and (7), we know the probability with which the one bit will be kept, and the probability with which the zero bits will be flipped to one. In case of RAPPOR, the one bit is kept with probability $\frac{e^{\varepsilon/2}}{e^{\varepsilon/2}+1}$ and each zero bit is flipped to one with probability $\frac{1}{e^{\varepsilon/2}+1}$. Then, denoting by N'_u the number of one bits in user u's reported bitvector B'_u, it follows that

$$\mathbb{E}[N'_u] = 1 \times \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}+1} + (|\mathcal{U}|-1) \times \frac{1}{e^{\varepsilon/2}+1} = \frac{e^{\varepsilon/2}+|\mathcal{U}|-1}{e^{\varepsilon/2}+1}. \tag{26}$$

Therefore, denoting the total number of one bits observed in all bitvectors reported across the whole population $\mathcal{P}$ by $N'_{\mathcal{P}}$:

$$\mathbb{E}[N'_{\mathcal{P}}] = \frac{|\mathcal{P}| \times (e^{\varepsilon/2}+|\mathcal{U}|-1)}{e^{\varepsilon/2}+1}. \tag{27}$$

On the other hand, in case of OUE, the one bit is kept with probability $\frac{1}{2}$ and each zero bit is flipped to one with probability $\frac{1}{e^{\varepsilon}+1}$. Thus, different from RAPPOR, $\mathbb{E}[N'_u]$ in OUE can be derived as

$$\mathbb{E}[N'_u] = 1 \times \frac{1}{2} + (|\mathcal{U}| - 1) \times \frac{1}{e^{\varepsilon}+1} = \frac{1}{2} + \frac{|\mathcal{U}|-1}{e^{\varepsilon}+1} = \frac{e^{\varepsilon} + 2|\mathcal{U}| - 1}{2 \times (e^{\varepsilon}+1)}. \tag{28}$$

Consequently, $\mathbb{E}[N'_{\mathcal{P}}]$ for OUE can be derived as

$$\mathbb{E}[N'_{\mathcal{P}}] = \frac{|\mathcal{P}| \times (e^{\varepsilon} + 2|\mathcal{U}| - 1)}{2 \times (e^{\varepsilon}+1)}. \tag{29}$$

Our BIAs for RAPPOR and OUE make use of the protocols' $\mathbb{E}[N'_{\mathcal{P}}]$. Having observed $\mathcal{O} = \{B'_1, B'_2, \ldots, B'_{|\mathcal{P}|}\}$, the adversary counts the total number of observed one bits across all bitvectors:

$$sum \leftarrow \sum_{i=1}^{|\mathcal{P}|} \sum_{j=1}^{|\mathcal{U}|} B'_i[j]. \tag{30}$$

Then, the adversary sets *sum* equal to $\mathbb{E}[N'_{\mathcal{P}}]$ (either Equation (27) for RAPPOR or Equation (29) for OUE) and solves for the only unknown, which is ε. The resulting ε becomes the adversary's prediction. The intuition behind this attack is that for a sufficiently large user population, the total number of observed one bits (the summation in Equation (30)) will converge to the mathematical estimate $\mathbb{E}[N'_{\mathcal{P}}]$. Thus, after counting the total number of one bits observed in $\mathcal{O}$, the adversary predicts ε as the value that is expected to yield this many one bits after data collection.

4.2 BIA for OLH

Recall from Section 2.2 that in OLH, users use hash functions H_u that map their value to an integer between 1 and g, i.e., $x_u \in [1, g]$. Afterward, randomized response is applied to x_u and the result of the randomized response is an integer $x'_u \in [1, g]$. Each user reports $\langle H_u, x'_u \rangle$.

Our BIA for OLH works as follows. The adversary observes $\mathcal{O} = \{\langle H_1, x'_1 \rangle, \langle H_2, x'_2 \rangle, \ldots, \langle H_{|\mathcal{P}|}, x'_{|\mathcal{P}|} \rangle\}$. Using $\mathcal{O}$, the adversary estimates the value of g as the difference between the maximum x' and the minimum x' that was observed in the user population:

$$g_{est} = \left(\max_{u \in \mathcal{P}} x'_u \right) - \left(\min_{u \in \mathcal{P}} x'_u \right). \tag{31}$$

For a reasonably large user population, we would expect $g_{est} = g$, i.e., the estimated g is equal to the real g. There are two reasons why the two may be different: (i) the user population is so small that $\mathcal{O}$ does not happen to contain all integers within $[1, g]$, (ii) there are too many hash collisions which cause the maximum integer g to never occur in $\mathcal{O}$. However, in practice and in our experiments, we observed that these two cases are highly unlikely. Hence, we generally have $g_{est} = g$.

Next, recall that in OLH, g is determined as: $g = \text{INT}(e^{\varepsilon} + 1)$. The adversary uses this fact to execute the BIA. That is, armed with g_{est}, the adversary sets $g_{est} = \text{INT}(e^{\varepsilon} + 1)$ and solves for the only unknown, which is ε. Here, notice that because of the integer rounding function $\text{INT}(\cdot)$, the adversary cannot obtain the ε value directly by solving: $\varepsilon \leftarrow \log(g_{est} - 1)$. To circumvent this, the adversary predicts ε as

$$\varepsilon \leftarrow \frac{\log(g_{est}) + \log(g_{est} - 1)}{2}. \tag{32}$$

4.3 BIA for SW

Recall from Equation (12) that SW has a parameter b which is directly linked to the value of ε. Similar to how we utilize the g parameter to solve for ε in our BIA for the OLH protocol, we utilize the b parameter of SW to solve for ε in our BIA for the SW protocol.

Our BIA for SW works as follows. The adversary observes $\mathcal{O} = \{v'_1, v'_2, \ldots, v'_{|\mathcal{P}|}\}$. Using $\mathcal{O}$, the adversary estimates the value of b as follows:

$$b_{est} = \frac{(\max_{u \in \mathcal{P}} v'_u) - (\min_{u \in \mathcal{P}} v'_u) - 1}{2}. \tag{33}$$

This is because by definition of the SW protocol, the value reported by each user is from the range $v'_u \in [-b, 1+b]$. Therefore, the total range of observed values (difference between max observed value and min observed value) is expected to be equal to $2b + 1$.

Next, armed with b_{est}, the adversary uses Equation (12) to solve for the only unknown, ε. Overall, combining all steps, the BIA can be stated as solving for ε in the following equality:

$$\frac{(\max_{u \in \mathcal{P}} v'_u) - (\min_{u \in \mathcal{P}} v'_u) - 1}{2} = \frac{\varepsilon e^{\varepsilon} - e^{\varepsilon} + 1}{2e^{\varepsilon}(e^{\varepsilon} - 1 - \varepsilon)}. \tag{34}$$

4.4 BIA for GRR

The GRR protocol does not have a parameter like g in OLH or b in SW which is linked to the privacy budget ε. Also, it is not possible to execute an attack based on the encoding type (like bitvector encoding used in RAPPOR or OUE) since GRR does not use an additional encoding, i.e., the inputs and outputs of GRR originate from the same $\mathcal{U}$. Hence, in the case of GRR, it is not possible to conduct a BIA without any knowledge of additional information about users' true values or f. In order to perform a BIA on GRR, we introduce an additional assumption: Although the uninformed adversary does not know f, we assume that the adversary knows $\mathcal{S} = \sum_{u \in \mathcal{P}} v_u$. Here, assume for simplicity that the domain $\mathcal{U}$ consists of integers; but if this is not the case, then each item can be trivially converted by performing a bijection (one-to-one mapping) between items in $\mathcal{U}$ and integers between $[1, |\mathcal{U}|]$. It is not possible to derive f from $\mathcal{S}$; therefore, the uninformed adversary still remains substantially weaker than the informed adversary.

Our BIA for GRR is presented in Algorithm 2, which is an adapted version of Algorithm 1 (hence, for brevity of the main article, it is given in Appendix B). While Algorithm 1 computes $\tilde{\text{f}}$ and uses the distance between f and $\tilde{\text{f}}$ as the *target*, the key difference is that our BIA for GRR computes $\tilde{\mathcal{S}}$ and uses the absolute value difference between $\mathcal{S}$ and $\tilde{\mathcal{S}}$ as the *target*. Then, similar to Algorithm 1, a search is performed to find the value of ε which would yield the data collection with the smallest resulting distance to *target*. The intuition behind this search is similar to Algorithm 1.

4.5 Theoretical Analysis

In this section, we theoretically analyze our uninformed BIAs for each protocol and show that the predicted budgets are unlikely to deviate from the real budgets with high probability.

Analysis for RAPPOR and OUE. In RAPPOR, since each user in $|\mathcal{P}|$ has 1 one bit and $|\mathcal{U}| - 1$ zero bits in their bitvector, $N'_{\mathcal{P}}$ in RAPPOR's BIA can be formulated as a sum of two Binomial random variables: (i) the first one follows a Binomial distribution with $|\mathcal{P}|$ trials and $\frac{e^{\varepsilon/2}}{e^{\varepsilon/2}+1}$ success probability, and (ii) the second one follows a Binomial distribution with $|\mathcal{P}| \times (|\mathcal{U}| - 1|)$ trials and $\frac{1}{e^{\varepsilon/2}+1}$ success probability. The expected value of $N'_{\mathcal{P}}$ was derived in Equation (27), and the observed *sum* that is used in the BIA is as shown in Equation (30).

OUE is also a bitvector-based protocol, and its BIA is almost identical to RAPPOR. Therefore, the analysis for OUE is similar to the analysis for RAPPOR. The main difference is the success probabilities of the Binomial random variables. $N'_{\mathcal{P}}$ in OUE's BIA can be formulated as a sum of two Binomial random variables: (i) the first one follows a Binomial distribution with $|\mathcal{P}|$ trials and $\frac{1}{2}$ success probability, and (ii) the second one follows a Binomial distribution with $|\mathcal{P}| \times (|\mathcal{U}-1|)$ trials and $\frac{1}{e^{\varepsilon}+1}$ success probability. The expected value of $N'_{\mathcal{P}}$ in OUE's case was derived in Equation (29), and the observed *sum* used in the BIA is the same as RAPPOR, i.e., it is shown in Equation (30).

We combine the above findings and use the fact that the two random variables in both RAPPOR and OUE's attack are independent. Then, using Hoeffding's inequality, we establish that

$$\Pr\left[|sum - \mathbb{E}[N'_{\mathcal{P}}]| \geq t\right] \leq 2 \times \exp\left(\frac{-2t^2}{|\mathcal{P}| \times |\mathcal{U}|}\right). \tag{35}$$

In other words, it is exponentially unlikely that the *sum* used in RAPPOR and OUE's BIAs will significantly deviate from $\mathbb{E}[N'_{\mathcal{P}}]$ as $|\mathcal{P}|$ and $|\mathcal{U}|$ get larger. This fact is also supported by our experiments, i.e., our results show that uninformed BIAs are very effective on RAPPOR and OUE, and furthermore, predicted budgets become more accurate as the population size and domain size are increased.

Analysis for OLH. In OLH, ε is predicted directly using g_{est}. Therefore, the effectiveness of the BIA is tightly linked with the correctness of g_{est} (Equation (31)). g_{est} will be correct if and only if both of the following conditions hold:

$$\left(\max_{u \in \mathcal{P}} x'_u\right) = g \qquad \text{and} \qquad \left(\min_{u \in \mathcal{P}} x'_u\right) = 1. \tag{36}$$

Assuming that the hash functions used in OLH are fair with respect to hash collisions, the probability that g_{est} is correct is

$$1 - 2 \times \left(\frac{g-1}{g}\right)^{|\mathcal{P}|} + \left(\frac{g-2}{g}\right)^{|\mathcal{P}|}. \tag{37}$$

For typical ε values between 0.1 and 5, the value of g remains less than 150. For small values of g such as $g \leq 10$, the probability that g_{est} is correct is close to 0.99, even for $|\mathcal{P}| = 50$. As g increases, the probability can become low for small populations (e.g., $|\mathcal{P}| \leq 200$), but for relatively large populations such as $|\mathcal{P}| \geq 1000$, the probability exceeds 0.99 even for large g.

Analysis for SW. In SW, the adversary exploits the fact that users' reported values should be within the range $[-b, 1+b]$. Using the reported values and this range, b_{est} is estimated, and ε is predicted using b_{est}. Therefore, the effectiveness of the BIA is tightly linked with the correctness of b_{est} (Equation (33)).

Since the output space of SW is continuous, the probability of observing the exact values $-b$ or $1+b$ is undefined. However, if at least one report lies within a small distance δ of $-b$ and $1+b$, the estimate b_{est} will be accurate within a factor of δ. Let $\eta = \frac{1}{2be^{\varepsilon}+1}$ be the constant tail density of the SW distribution. The probability that a single report v'_u lies in the left or right tail interval of width δ is

$$\Pr[v'_u \in [-b, -b+\delta]] = \eta \cdot \delta, \quad \Pr[v'_u \in [1+b-\delta, 1+b]] = \eta \cdot \delta. \tag{38}$$

Then, the probability that none of the $|\mathcal{P}|$ users report a value in either tail region is

$$\Pr[\text{fail}] \leq 2 \cdot (1 - \eta \cdot \delta)^{|\mathcal{P}|}. \tag{39}$$

Hence, the adversary's probability in successfully estimating b_{est} within a small δ satisfies:

$$\Pr[\text{success}] \geq 1 - 2 \cdot \left(1 - \frac{\delta}{2be^{\varepsilon}+1}\right)^{|\mathcal{P}|}. \tag{40}$$

This bound demonstrates that the adversary can accurately estimate b (and thus ε) with high probability as long as the population size $|\mathcal{P}|$ is sufficiently large. As $|\mathcal{P}|$ increases, the success probability converges exponentially to 1. For example, for $\delta = 0.01$ and moderate values of b and ε, values of $|\mathcal{P}|$ close to 10^4 typically ensure reliable inference.

5 Experimental Evaluation

5.1 Experiment Setup and Datasets

We implemented all LDP protocols and BIAs in Python. In this section, we report the experimental evaluations of our informed and uninformed BIAs using five datasets, three distance metrics, five LDP protocols, and varying ε budgets that are commonly used in the literature.

Datasets. We used five datasets that are also commonly used in the DP and LDP literatures for experimentation: Adult, Kosarak, MSNBC, Census, and Zipfian.

Adult was originally extracted from the Census database. It contains 45.222 records with 14 attributes relating to individuals' census information such as age, education, work class, marital status, race, and gender. We downloaded the Adult dataset from the UCI Machine Learning Repository[2] and used the individuals' *age* values in our experiments. Ages in the dataset ranged between 17 and 90, therefore, we have $|\mathcal{U}| = 74$.

Kosarak contains anonymized click-stream data of a Hungarian online news portal. We downloaded it from the FIMI Dataset Repository.[3] Each row in the dataset corresponds to one user's URL visit sequence. Due to many URLs having very few occurrences (e.g., one or two occurrences), we pre-processed the dataset by identifying the top-128 most visited URLs and removed the rest. Consequently, we have $|\mathcal{U}| = 128$. For users who had more than one URL in their resulting stream, the most frequently occurring URL in their stream was picked as their v_u. Finally, we sampled $|\mathcal{P}| = 500.000$ users from the dataset to be used in our experiments.

MSNBC contains browsing activity logs of individuals who accessed msnbc.com on September 28, 1999. We downloaded the MSNBC dataset from the UCI Machine Learning Repository.[4] Each row in the dataset represents a user's sequence of page visits, but different from Kosarak, page visits are recorded by category (e.g., news, tech, weather, and sports) rather than URL. There are $|\mathcal{U}| = 17$ categories. For each individual, we used the first entry in their row (i.e., the first page category they visited) as their v_u. The cardinality of the population is $|\mathcal{P}| = 989.818$.

Census is curated from the 2019 US Census Bureau Database, and it was also used in [27]. It is similar to the Adult dataset. The dataset was obtained from the IPUMS interface and includes records from the Annual Social and Economic Supplement. Each record represents one individual and contains demographic and employment-related attributes, e.g., age, gender, race, marital status, and education. The population size is $|\mathcal{P}|$ = 1,676,013.

Finally, *Zipfian* is a synthetic dataset in which users' true values follow a Zipfian distribution with skewness parameter $s = 1.5$ and population size $|\mathcal{P}| = 100,000$. For our experiments, we generate multiple versions of the Zipfian dataset with varying domain sizes $|\mathcal{U}|$ = 8, 16, 32, 64, 128, and 256.

Error Metrics. The goal of our BIAs is to accurately estimate the privacy budget used in LDP data collection. Let ε^* denote the true privacy budget, and let $\hat{\varepsilon}$ denote the budget predicted by our BIA. To account for the randomness in LDP, we repeat each experiment $n = 10$ times and average the results. We use the subscript $\hat{\varepsilon}_i$ to denote the prediction made by our BIA in the i'th experiment.

[2] https://archive.ics.uci.edu/dataset/2/adult
[3] http://fimi.uantwerpen.be/data/
[4] https://archive.ics.uci.edu/dataset/133/msnbc+com+anonymous+web+data

Table 1. MAEs of Our BIAs for Informed Adversaries Using Three Datasets (Adult, Kosarak, MSNBC), Five LDP Protocols (GRR, OUE, RAPPOR, OLH, SW), and Varying Real Budgets ε^* between 0.1 and 3.0

Real ε^*	**Adult**					**Kosarak**					**MSNBC**				
	GRR	**OUE**	**RAP**	**OLH**	**SW**	**GRR**	**OUE**	**RAP**	**OLH**	**SW**	**GRR**	**OUE**	**RAP**	**OLH**	**SW**
$\varepsilon^* = 0.1$	<0.01	<0.01	0.05	0.01	<0.01	0.06	0.01	<0.01	0.01	1.0	0.02	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 0.5$	0.05	0.03	0.09	0.18	0.01	0.02	<0.01	0.01	0.01	0.55	0.01	<0.01	0.01	<0.01	0.22
$\varepsilon^* = 1.0$	0.14	0.03	0.05	0.17	0.04	0.05	<0.01	<0.01	<0.01	0.72	0.01	<0.01	<0.01	<0.01	0.67
$\varepsilon^* = 1.5$	0.16	0.01	0.08	0.01	0.02	<0.01	<0.01	<0.01	<0.01	0.02	0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 2.0$	0.13	0.05	0.05	0.02	0.06	<0.01	<0.01	<0.01	<0.01	0.02	0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 2.5$	0.02	0.02	0.04	0.03	0.07	0.01	<0.01	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	<0.01	0.01
$\varepsilon^* = 3.0$	0.01	0.01	0.03	0.02	0.03	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	0.01	<0.01	<0.01

RAPPOR is abbreviated as RAP. MAEs lower than 0.01 after rounding are denoted by "< 0.01" throughout the table (instead of 0).

Table 2. MSEs of Our BIAs for Informed Adversaries Using Three Datasets (Adult, Kosarak, MSNBC), Five LDP Protocols (GRR, OUE, RAPPOR, OLH, SW), and Varying Real Budgets ε^* between 0.1 and 3.0

Real ε^*	**Adult**					**Kosarak**					**MSNBC**				
	GRR	**OUE**	**RAP**	**OLH**	**SW**	**GRR**	**OUE**	**RAP**	**OLH**	**SW**	**GRR**	**OUE**	**RAP**	**OLH**	**SW**
$\varepsilon^* = 0.1$	0.12	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	0.92	<0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 0.5$	0.09	0.01	0.01	0.01	<0.01	<0.01	<0.01	<0.01	<0.01	0.37	<0.01	<0.01	<0.01	<0.01	0.01
$\varepsilon^* = 1.0$	0.10	<0.01	0.02	0.01	<0.01	<0.01	<0.01	<0.01	<0.01	0.07	<0.01	<0.01	<0.01	<0.01	0.45
$\varepsilon^* = 1.5$	0.01	<0.01	<0.01	0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 2.0$	<0.01	<0.01	0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 2.5$	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 3.0$	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01

MSEs lower than 0.01 after rounding are denoted by "< 0.01" throughout the table (instead of 0).

To evaluate the performance of our attacks, we utilize two error metrics: MAE and **Mean Squared Error** (**MSE**). The *MAE* captures the average magnitude of the prediction errors and is defined as

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\varepsilon^* - \hat{\varepsilon}_i|. \tag{41}$$

The *MSE* also measures the average error but penalizes larger deviations more heavily by squaring each error term:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\varepsilon^* - \hat{\varepsilon}_i)^2. \tag{42}$$

5.2 Results of BIAs for Informed Adversaries

We first evaluate our BIAs for informed adversaries (i.e., attacks from Section 3). We report both the MAE and MSE results of our BIAs on three datasets, five LDP protocols, and varying ε^* in Tables 1 and 2, respectively. We generally observe that our BIAs are highly effective. According to the MAE table, in more than 73% of the reported results, the MAEs are strictly lower than 0.03. Furthermore, in approximately 50% of the reported results, the MAEs are strictly lower than 0.01. This effectiveness is corroborated by the MSE results, where 83% of reported values are below 0.01, further confirming the robustness of our BIAs against informed adversaries. These results show that the adversary can predict the privacy budget with very little error using our BIAs.

Table 3. MAEs of Our BIAs for Uninformed Adversaries Using Three Datasets, Five LDP Protocols, and Varying Real Budgets ε^*

Real ε^*	Adult					Kosarak					MSNBC				
	GRR	OUE	RAP	OLH	SW	GRR	OUE	RAP	OLH	SW	GRR	OUE	RAP	OLH	SW
$\varepsilon^* = 0.1$	0.1	<0.01	<0.01	0.24	<0.01	0.28	<0.01	<0.01	0.24	<0.01	<0.01	<0.01	<0.01	0.24	<0.01
$\varepsilon^* = 0.5$	0.29	<0.01	<0.01	0.39	<0.01	0.04	<0.01	<0.01	0.39	<0.01	0.01	<0.01	<0.01	0.39	<0.01
$\varepsilon^* = 1.0$	0.22	<0.01	<0.01	0.24	<0.01	0.07	<0.01	<0.01	0.24	<0.01	0.02	<0.01	<0.01	0.24	<0.01
$\varepsilon^* = 1.5$	0.04	<0.01	0.01	<0.01	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 2.0$	0.04	<0.01	0.01	0.01	<0.01	0.01	<0.01	<0.01	0.01	<0.01	<0.01	<0.01	<0.01	0.01	<0.01
$\varepsilon^* = 2.5$	0.15	<0.01	<0.01	0.02	<0.01	0.02	<0.01	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	0.02	<0.01
$\varepsilon^* = 3.0$	0.07	<0.01	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	0.02	<0.01

Across the different datasets, we observe from Table 1 that our informed BIAs usually perform better on RAPPOR and OUE protocols, whereas they perform relatively worse on GRR and SW protocols. For the OLH protocol, BIAs perform well (close to OUE and RAPPOR) on Kosarak and MSNBC datasets, whereas they perform relatively worse on the Adult dataset. In the case of GRR, this performance decrease stems from a fundamental characteristic of the GRR protocol: when an input value v_u undergoes perturbation, the resulting output y_u loses any meaningful information about the original input value when $y_u \neq v_u$. This is because GRR samples a fake item uniformly at random when $y_u \neq v_u$. This loss of information significantly worsens the estimation process, and therefore the Aggregate followed by Calc_Distance steps in Algorithm 1 may produce distances that contain relatively high noise. The increased utility loss of GRR, especially with large $|\mathcal{U}|$, was also highlighted in relevant works [10, 23, 54]. On the other hand, protocols such as RAPPOR and OUE do not suffer from the same problem, which fuels our ability to achieve higher BIAs. In the case of SW, we observe that its high MAEs and MSEs occur when $\varepsilon \leq 1$ and on the Kosarak and MSNBC datasets. Typically, the SW protocol tends to generate estimations that closely approximate the real distributions; however, in these specific instances, SW's estimations resulting from maximum likelihood estimation tend to contain higher error. As a consequence, the aggregation and distance computations of Algorithm 1 are more noisy, and therefore the errors of our BIAs increase.

It is also worth noting that BIAs on the MSNBC dataset are generally more accurate than on Adult and Kosarak, and BIAs on the Adult dataset are generally less accurate than on Kosarak and MSNBC. This is because the MSNBC dataset has the largest $|\mathcal{P}|$ and the smallest $|\mathcal{U}|$. Hence, LDP frequency estimations are likely to be most accurate on the MSNBC dataset. In contrast, Adult has the smallest $|\mathcal{P}|$, and it is known that LDP yields more noisy frequency estimations when $|\mathcal{P}|$ is small [23]. Overall, we arrive at the conclusion that the more accurate the estimations are, the more accurate our BIAs will be.

5.3 Results of BIAs for Uninformed Adversaries

In this section, we evaluate our BIAs for uninformed adversaries, i.e., attacks from Section 4. We report the MAE and MSE results in Tables 3 and 4, respectively. Again, the results show that our BIAs are highly effective, with 83% of the MAEs being strictly lower than 0.03 and 89% of the MSEs being strictly lower than 0.01. Comparing the results in Tables 1 and 2 with Tables 3 and 4 allows us to also analyze whether informed BIAs are more or less effective compared to uninformed BIAs. For the GRR protocol, we observe that informed BIAs are generally more effective than uninformed BIAs. For the RAPPOR and OUE protocols, we observe that informed and uninformed BIAs have similar levels of effectiveness, as both typically achieve very low MAEs and MSEs. For the OLH protocol, we again observe that informed BIAs are more effective than uninformed BIAs, especially

Table 4. MSEs of Our BIAs for Uninformed Adversaries Using Three Datasets, Five LDP Protocols, and Varying Real Budgets ε^*

Real ε^*	Adult					Kosarak					MSNBC				
	GRR	OUE	RAP	OLH	SW	GRR	OUE	RAP	OLH	SW	GRR	OUE	RAP	OLH	SW
$\varepsilon^* = 0.1$	<0.01	<0.01	<0.01	0.06	<0.01	0.03	<0.01	<0.01	0.06	<0.01	<0.01	<0.01	<0.01	0.06	<0.01
$\varepsilon^* = 0.5$	0.07	<0.01	<0.01	0.15	<0.01	0.08	<0.01	<0.01	0.15	<0.01	<0.01	<0.01	<0.01	0.15	<0.01
$\varepsilon^* = 1.0$	0.03	<0.01	<0.01	0.05	<0.01	<0.01	<0.01	<0.01	0.05	<0.01	<0.01	<0.01	<0.01	0.05	<0.01
$\varepsilon^* = 1.5$	0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 2.0$	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 2.5$	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
$\varepsilon^* = 3.0$	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01

for low ε^*. These results show that the availability of f for the informed adversary contributes positively to executing BIAs with higher effectiveness. Yet, the SW protocol is an exception. For SW, uninformed BIAs consistently achieve lower MAEs and MSEs than informed BIAs. Hence, in the case of SW, it is in the best interest of the adversary to execute the uninformed BIA rather than the informed BIA, even if they have access to f. Next, we analyze and discuss the different protocols individually.

Across all protocols, datasets, and ε^* values, our uninformed BIAs for RAPPOR and OUE achieve both MAEs and MSEs ≤ 0.01. This shows that our attack strategy based on counting the number of one bits in the reported bitvectors is highly effective in practice. We also observe that the MAEs and MSEs on Kosarak and MSNBC datasets are slightly lower than those on the Adult dataset, which is intuitive because Kosarak and MSNBC's population sizes are larger. When population sizes are larger, the law of large numbers implies that the total number of observed one bits is more likely to converge to $\mathbb{E}[N'_{\mathcal{P}}]$.

In the OLH protocol, we observe that the uninformed BIAs perform well when $\varepsilon^* \geq 1.5$, but they perform worse when $\varepsilon^* \leq 1.0$. This is due to how the g parameter is determined in OLH, which is pivotal for our BIA since it utilizes g. Between $\varepsilon^* = 0.1$ and $\varepsilon^* = 1.0$, the function $g = \text{INT}(e^{\varepsilon^*} + 1)$ grows very slowly, leading to several values of ε^* mapping to the same or similar g. Hence, observing g_{est} does not reveal the budget precisely, which results in higher MAEs and MSEs. On the other hand, for $\varepsilon^* \geq 1.5$, g grows faster, and changes in g_{est} correspond more directly to changes in ε^*. This stronger correlation yields lower MAEs and MSEs. Additionally, we observe that the MAE and MSE values for OLH are identical across the three datasets, as our BIA deterministically identifies g_{est} by computing $\max x'_u$ and $\min x'_u$ across $u \in \mathcal{P}$. This leads to the same budget estimates due to Equation (32) and thus identical errors.

Analyzing the GRR protocol, we observe that our BIAs perform best on the MSNBC dataset, followed by Kosarak, with the Adult dataset yielding the highest MAEs and MSEs. The intuition lies in our BIA's use of $\mathcal{S}$ and the population size $|\mathcal{P}|$. When the domain size $|\mathcal{U}|$ is small, the knowledge of $\mathcal{S}$ becomes more valuable. As a result, generating $\mathcal{P}_{\text{syn}}$ with the same sum $\mathcal{S}$ yields a synthetic value distribution that more closely resembles the true one in $\mathcal{P}$, improving the BIA's accuracy. MSNBC, which has the smallest $|\mathcal{U}|$ and largest $|\mathcal{P}|$, achieves the lowest errors. In contrast, the Adult dataset—with the largest $|\mathcal{U}|$ and smallest $|\mathcal{P}|$—yields the highest errors.

Finally, we analyze the results of our BIAs for the SW protocol. As opposed to the previous protocols, the SW protocol is an exception since uninformed BIAs in SW perform better than informed BIAs. Results show that uninformed BIAs achieve both MAEs and MSEs < 0.01 across all ε^* and datasets. The success of our BIAs implies that the attack's estimation of the b parameter (denoted by b_{est}) is highly accurate, which indicates that the maximum and minimum possible v'_u

Table 5. Comparison of MAEs of Our Informed BIAs Using Three Different Distance Metrics in Calc_Distance: ℓ_1 distance, ℓ_2 Distance, and KL Divergence

		ℓ_1 distance	ℓ_2 distance	KL divergence
Adult	$\varepsilon^* = 0.1$	<0.01	0.04	0.04
	$\varepsilon^* = 0.5$	0.03	0.08	0.05
	$\varepsilon^* = 1.0$	0.03	0.04	0.03
	$\varepsilon^* = 1.5$	0.01	0.02	0.04
	$\varepsilon^* = 2.0$	0.05	0.03	0.02
	$\varepsilon^* = 2.5$	0.02	0.05	0.02
	$\varepsilon^* = 3.0$	0.01	0.01	0.01

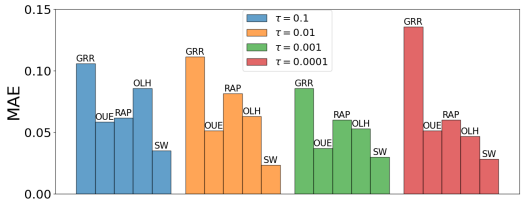


Fig. 1. Impact of the τ parameter on MAEs of our informed BIAs (Adult dataset).

values are indeed commonly observed in $\mathcal{O}$. The fact that Equation (34) directly links ε^* with b_{est} without any integer rounding (such as the rounding in OLH, which reduces BIA effectiveness) also contributes positively to the effectiveness of BIAs in SW.

5.4 Analysis of Attack Settings and Parameters

Impact of the distance metric. Recall from Section 3.2 that the Calc_Distance function can use three distance metrics: ℓ_1 distance, ℓ_2 distance, and KL divergence. In Table 5, we compare the MAEs of our attacks using each of these three distance metrics to evaluate how the choice of distance metric impacts attack accuracy. We only provide the results on the Adult dataset since MAEs on the other datasets are much lower and much more similar to one another; therefore, they do not enable a meaningful comparison. We observe from the results that in all ε^* values other than $\varepsilon^* = 2.0$, ℓ_1 distance yields the lowest MAE. Thus, we use ℓ_1 distance as our distance metric by default throughout the article. Yet, we also observe that our BIAs are not too highly affected by the choice of the distance metric. In other words, although ℓ_1 distance seems to be a good choice, MAEs with other metrics are not too different. This is a positive result, which shows that our BIAs are not too sensitive to the choice of distance metric.

Impact of τ. Recall from Section 3.1 that Algorithm 1 uses a stopping condition parameter τ which affects when the algorithm will terminate. The τ parameter, therefore, has two impacts: (i) it affects MAE since early termination may result in less accurate predictions, and (ii) it affects the attack's total execution time since a larger number of iterations increases the execution time. In Figure 1 and Table 6, we plot the MAEs and execution times of varying τ between 0.1 and 0.0001.

As expected, we observe from Table 6 that reducing the τ value increases the total execution time. This is because lower τ implies more iterations of our BIA, and within each iteration, an LDP simulation is performed with the current budget ε_{mid}. Hence, a larger number of iterations directly increases the execution time. However, an interesting observation is made from Figure 1—reducing the τ value does not necessarily decrease the MAEs of our attack. Particularly, the predicted budgets

Table 6. Impact of the τ Parameter on the Total Execution Times of Our Informed BIAs (Total of All Five Protocols, Adult Dataset is Used)

Value of τ	Execution Time
$\tau = 0.1$	13.7 minutes
$\tau = 0.01$	20.1 minutes
$\tau = 0.001$	30.8 minutes
$\tau = 0.0001$	36.7 minutes

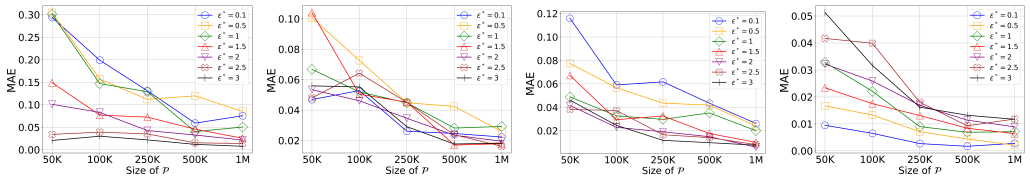


Fig. 2. Impact of population size $|\mathcal{P}|$ on MAE for varying ε^* and protocols (from left to right: GRR, RAPPOR, OUE, SW). For each experiment, we randomly sample subsets of size 50.000, 100.000, 250.000, 500.000, and 1.000.000 from the Census dataset.

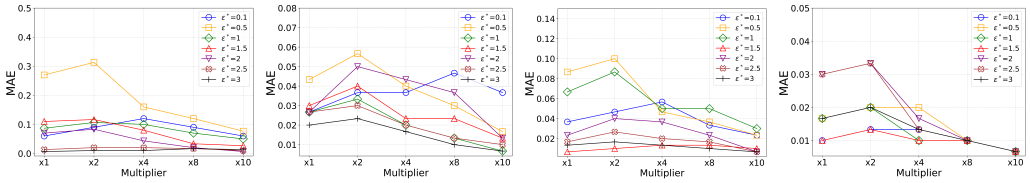


Fig. 3. Impact of the population size $|\mathcal{P}|$ on MAE for varying ε^* and protocols (from left to right: GRR, RAPPOR, OLH, SW). To simulate varying $|\mathcal{P}|$ while keeping other factors constant, records in the Adult dataset are multiplied (x1, x2, x4, x8, x10) where x1 denotes the original dataset.

in GRR have the highest MAE when $\tau = 0.0001$. For protocols other than GRR, we observe that the MAEs are relatively stable regardless of τ varying between 0.1 and 0.0001. For several protocols, slight MAE decreases can be observed as τ is reduced, which is intuitive.

As stated in Section 3.1, we used $\tau = 0.001$ as the default value of τ throughout the article. The results in Figure 1 and Table 6 support this choice, considering that the MAEs of all protocols are generally low when $\tau = 0.001$. Furthermore, when executed on a commodity laptop, the total execution time of our BIAs for all five protocols combined remains close to 30 minutes. Hence, we conclude that $\tau = 0.001$ yields good MAEs in a reasonable amount of time.

5.5 Impact of Population Size

Next, we analyze the impact of the user population size $|\mathcal{P}|$ on the effectiveness of our BIA, combining both real and synthetically amplified datasets. To isolate the effect of $|\mathcal{P}|$ while keeping other factors constant, we consider two approaches. First, we randomly sampled subsets of different sizes (50,000 to 1,000,000) from the Census dataset to reflect realistic population variability without duplication. Second, we synthetically amplified the Adult dataset by duplicating each record multiple times, creating larger populations (e.g., x2, x4, ..., x10). In both cases, MAE results under varying real budgets ε^* and protocols are presented in Figures 2 and 3.

Results of these experiments reveal a consistent trend: BIAs are generally more effective as $|\mathcal{P}|$ increases. Although the reduction in MAEs is not always monotonic–owing to the probabilistic

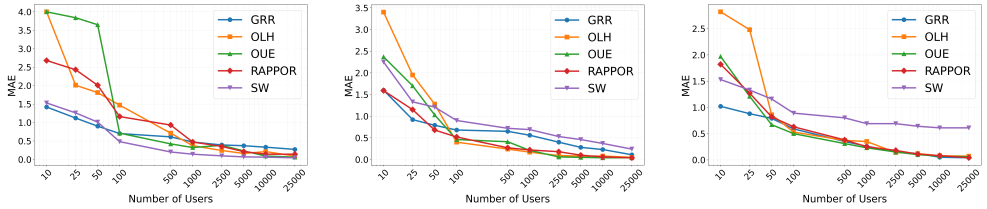


Fig. 4. Impact of small population sizes $|\mathcal{P}|$ on MAE for varying protocols and datasets (from left to right: Adult, Kosarak, MSNBC) for informed adversaries. To simulate small $|\mathcal{P}|$, users are randomly selected from the entire population in each iteration, and the selected population uses $\varepsilon^* = 1$.

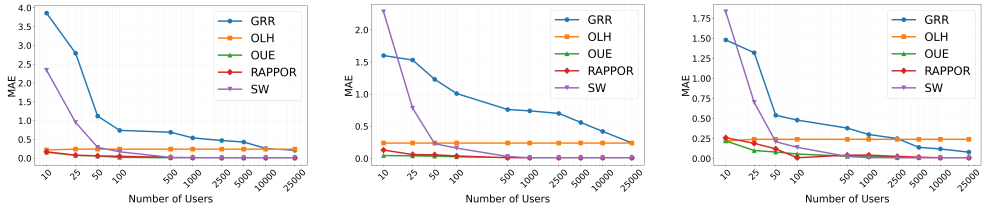


Fig. 5. Impact of small population sizes $|\mathcal{P}|$ on MAE for varying protocols and datasets (from left to right: Adult, Kosarak, MSNBC) for uninformed adversaries. To simulate small $|\mathcal{P}|$, users are randomly selected from the entire population in each iteration, and the selected population uses $\varepsilon^* = 1$.

nature of both the protocols and our BIAs–the overall trend holds. The increase in effectiveness stems from two key factors. First, a larger $|\mathcal{P}|$ provides the adversary with a larger $\mathcal{O}$, offering more data points for the attack. Second, LDP protocols produce more accurate frequency estimates with a larger population. These improvements enhance the accuracy of both the adversary's reconstructed distribution $\tilde{\mathrm{f}}$ and the simulated distribution $\mathrm{f}_{\mathrm{syn}}$, leading to more effective BIAs.

In the next set of experiments, motivated by the notion of PLDP in which different users may use different budgets and, therefore, the number of users using the same budget can be quite few, we reduced the population size $|\mathcal{P}|$ substantially. In particular, we experimented with very low values of $|\mathcal{P}|$ such as $|\mathcal{P}| = 10, 25, 50, 100$ which reflects PLDP, e.g., users can choose different ε^* and very few of them (such as 10, 25, 50, 100) happen to use the same ε^*. In addition, we also experimented with values of $|\mathcal{P}|$ such as $|\mathcal{P}| = 500, 1{,}000, 5{,}000$, and so on which can reflect scenarios similar to PLDP, e.g., users are given different ε^* options to choose from, and 500, 1,000, or 5,000 users happen to choose the same ε^* option. We ran these experiments both with informed and uninformed adversaries. Results with informed adversaries are shown in Figure 4 and results with uninformed adversaries are shown in Figure 5.

As shown in Figure 4, MAEs are highest when $|\mathcal{P}|$ is small, e.g., between 10–100. It can be concluded that the limited size of $\mathcal{O}$, which may occur in PLDP scenarios, is indeed a cause for higher error in informed BIAs. However, as the number of users increases, MAEs decrease and converge towards the MAE values attained in Table 1, demonstrating the stabilization effect of larger population sizes on budget inference. It can also be observed from Figure 4 that different protocols are affected differently by the number of users. In general, OLH is most badly affected, as its MAEs are high for small $|\mathcal{P}|$. In contrast, GRR is not as badly affected by the reduction in $|\mathcal{P}|$. The behaviors of SW, RAPPOR, and OUE are dataset-dependent, since different datasets have different $\mathcal{U}$.

For the uninformed adversary case, we observe distinct trends across the protocols. As shown in Figure 5, MAEs of OUE, RAPPOR, and OLH remain largely constant across all population sizes. This behavior is due to the nature of the BIA for OUE and RAPPOR, which allows the adversary to

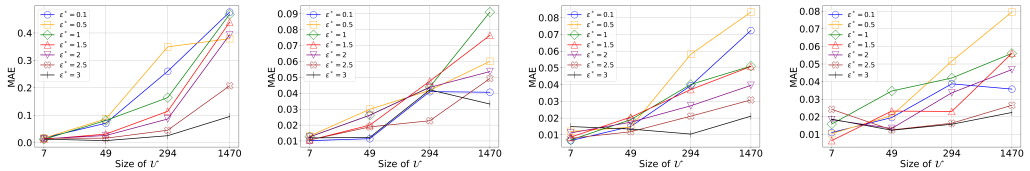


Fig. 6. MAE results of informed BIAs across four LDP protocols—GRR, RAPPOR, OUE, and OLH (left to right)—on the cross product of multiple attributes: `Workclass`, `Marital-status`, `Relationship`, and `Race` from the Adult dataset.

estimate ε^* with small error. For OLH, as long as there are sufficient reports for the adversary to identify the maximum and minimum observed values (x'_u), the primary source of error stems from the logarithmic calculation process of g_{est}. Additionally, since ε^* is set to 1 in these experiments, it becomes easier for the adversary to observe these values accurately. In contrast, for GRR and SW, MAEs follow a decreasing trend similar to the informed adversary case. For SW, this is attributed to the broader interval for user values when $\varepsilon^* = 1$, making it more challenging for the adversary to observe the maximum (v'_u) and minimum (v'_u) values to accurately estimate b_{est}. For GRR, as the population size increases, the adversary's constructed population through summation aligns more closely with the original population, leading to a reduction in MAE.

Comparing Figures 4 and 5, we observe that the reduction in $|\mathcal{P}|$ more heavily affects the informed adversary than the uninformed adversary. This is because $\tilde{\mathrm{f}}$ in Algorithm 1 is a key factor in the accuracy of the informed adversary. When $|\mathcal{P}|$ is small, $\tilde{\mathrm{f}}$ is more random, i.e., it does not yield an accurate representation of the LDP protocol's behavior. This causes the search process of Algorithm 1 to also be inaccurate. On the other hand, the uninformed adversary does not use $\tilde{\mathrm{f}}$. Instead, the uninformed adversary uses simple statistics such as $N'_{\mathcal{P}}$, g_{est}, and b_{est}. Despite small population sizes, retrieving these values is usually easier. For example, GRR's MAEs in Figure 5 are typically higher than the other protocols, which can be attributed to the fact that its inference process under an uninformed adversary is closest to that of an informed adversary. Yet, in Figure 5, many protocols' MAEs remain low despite very small populations. This is a positive result for our BIAs since they demonstrate that our BIAs can remain effective under PLDP scenarios as well.

5.6 Impact of Domain Size

In this section, we analyze the impact of the domain size $|\mathcal{U}|$ on the effectiveness of our BIAs. We perform experiments in two categories: (i) experiments in which multiple attributes (columns) are simultaneously perturbed with LDP, and (ii) experiments with varying $|\mathcal{U}|$.

Perturbing multiple attributes. Recall from Section 5.1 that Adult is a tabular dataset that contains multiple attributes. Here, we consider the case in which LDP is applied to multiple attributes simultaneously. In this case, we treat $\mathcal{U}$ as the cross product of the individual attributes. From the Adult dataset, we pick the following attributes: `Workclass`, `Marital-status`, `Relationship`, and `Race`, which have individual domain sizes of 7, 7, 6, and 5, respectively. We begin by perturbing `Workclass` attribute alone, such that $|\mathcal{U}| = 7$. We then progressively expand to the cross product of the other attributes, first with `Marital-status` (domain size becomes $|\mathcal{U}| = 7 \times 7 = 49$), then `Relationship` (domain size becomes $|\mathcal{U}| = 294$), and finally, `Race` (domain size becomes $|\mathcal{U}| = 1,470$). This approach allows us to observe the impact of increasing data dimensionality and $|\mathcal{U}|$ on the performance of our BIAs.

Results of our BIAs are shown in Figure 6 with plots for GRR, RAPPOR, OUE, and OLH protocols arranged from left to right. The results show a consistent trend: as more attributes are combined via cross-product and the domain size grows, the MAEs of our BIAs increase. This is intuitive,

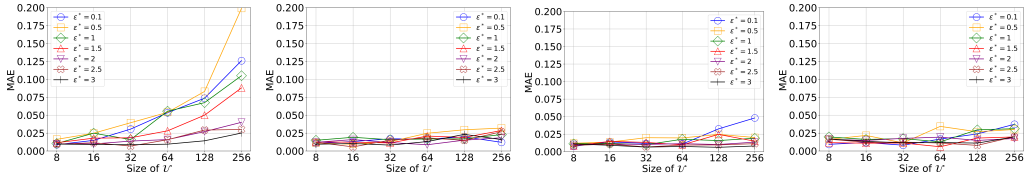


Fig. 7. MAE results of informed BIAs across four LDP protocols—GRR, RAPPOR, OUE, and OLH (left to right)—on Zipfian datasets where the domain size is varied between 8 and 256.

as larger domain sizes reduce the frequency of individual domain elements, making the inference task more challenging. The effect is especially pronounced in the case of GRR. While the increase in MAE for RAPPOR, OUE, and OLH is moderate, i.e., typically around 0.08, the GRR protocol exhibits a substantial rise in error, with the MAE increasing by nearly 0.5. This increase can be linked to the nature of the GRR mechanism, which struggles to preserve utility when the domain size $|\mathcal{U}|$ is large. In general, the utility of the remaining protocols is less affected by $|\mathcal{U}|$ compared to GRR.

We highlight that the increasing trend in MAEs is also consistent with the theoretical analysis we perform in Section 3.3. According to the theoretical analysis, $|\mathcal{U}|$ and $|\mathcal{P}|$ affect the effectiveness of BIAs in reverse. As we observed in the previous section, increasing $|\mathcal{P}|$ causes reduced MAEs. In contrast, we observe in this section that increasing $|\mathcal{U}|$ causes increased MAEs, which fits the intuition from the theoretical analysis. We note that despite the increase in error, results show that the proposed BIAs remain effective even in the multi-attribute setting. In all protocols, the inferred budgets remain close to the true budgets, e.g., MAEs on RAPPOR, OUE, and OLH protocols all remain lower than 0.1 throughout Figure 6. This demonstrates that adversaries can still mount successful BIAs when users report multiple attributes under LDP.

Experiments with varying $|\mathcal{U}|$. We utilize synthetic Zipfian datasets to further evaluate the impact of domain size on the effectiveness of our BIAs. Since the datasets are synthetic, we are able to generate datasets with varying domain sizes $|\mathcal{U}| = 8$, 16, 32, 64, 128, and 256. Unlike the multiple attribute setting discussed above, where increasing the domain size through cross product also changes the underlying statistical distribution of the data, here the overall shape of the distribution remains fixed to Zipfian across varying $|\mathcal{U}|$. This means that even as the domain grows, a small number of elements continue to dominate the frequency distribution, preserving the heavy-tailed nature of the data.

Figure 7 presents the MAE results for the GRR, RAPPOR, OUE, and OLH protocols, respectively. As expected, the GRR protocol exhibits increasing MAEs as $|\mathcal{U}|$ grows, due to its inherent sensitivity to large domain sizes. In contrast, the other protocols—RAPPOR, OUE, and OLH—demonstrate relatively stable error rates, with only minor increases in MAEs that become observable as $|\mathcal{U}| = 128$ and 256. Some minor fluctuations in MAEs for smaller $|\mathcal{U}|$ are also attributable to the randomized nature of LDP. We observe that these findings align with the earlier findings reported in Figure 6, in which higher MAEs were observed after $|\mathcal{U}| \geq 204$. Altogether, these results highlight that the impact of domain size on BIA effectiveness is protocol-dependent (e.g., more significant in GRR and less significant in other protocols), and starts to become noticeable only after $|\mathcal{U}| \geq 100$ or 200.

5.7 Using BIAs to Increase the Effectiveness of Other Attacks

Finally, in this section, we show how budget inference can be used by an adversary to increase their performance in another attack, namely, the MGA attack [7]. MGA is a poisoning attack. In poisoning attacks, the adversary injects malicious data (e.g., through the participation of adversary-controlled user devices in the LDP data collection process) to distort the estimated statistics on the server side.

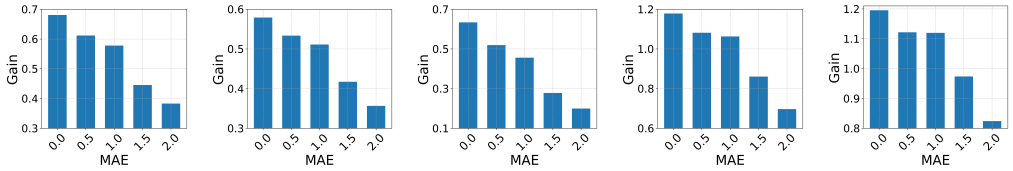


Fig. 8. Gains of poisoning attacks (MGA) under varying MAE. From left to right, datasets are: Adult, Census, MSNBC, Zipfian with $|\mathcal{U}| = 64$, and Zipfian with $|\mathcal{U}| = 128$.

Recall that $\mathrm{f} = \{f_v\}_{v \in \mathcal{U}}$ denotes the true frequency distribution of the items in the domain. Let $\mathrm{f}' = \{f'_v\}_{v \in \mathcal{U}}$ denote the frequencies after a poisoning attack such as MGA. The adversary's Gain from the poisoning attack is formulated as

$$\text{Gain} = \sum_{v \in \mathcal{U}} |f_v - f'_v|. \tag{43}$$

We create an experimental scenario in which the OLH protocol is used for data collection, and the real budget is $\varepsilon^* = 1.0$. The adversary injects $10\% \times |\mathcal{P}|$ malicious users in the LDP data collection process. These malicious users use MGA [7] to maximize the Gain of the poisoning attack. To demonstrate how our BIAs can provide benefit to the poisoning attack, we assume that the adversary does not know the value of ε^*. Instead, the adversary performs attacks assuming different budgets = 1.0, 1.5, 2.0, 2.5, and 3.0. Compared to the real budget ε^*, these correspond to MAE = 0, 0.5, 1.0, 1.5, and 2.0. Under each of these scenarios, we measure the Gain achieved by the adversary. The results are reported in Figure 8 for multiple datasets.

Results show that, across multiple datasets, MAE = 0 is the case when the adversary's Gain is the highest. As MAE increases, the adversary's Gain decreases. In other words, the lower the MAE, i.e., the more accurately the adversary knows the privacy budget, the more successful the poisoning attack will be. Consequently, an adversary who wishes to perform a successful poisoning attack may want to use the BIAs proposed in this article to first infer the budget and then perform the poisoning attack accordingly, in order to improve the Gain of their poisoning attack. The reason why accurate knowledge of the budget improves the poisoning attack is because the poisoning attack aims to find the best $\langle H_u, x'_u \rangle$ that increases the Gain. If the adversary knows the ε budget, since g (i.e., the output space of H_u) is controlled by g, the adversary can also find a better $\langle H_u, x'_u \rangle$ pair that yields higher Gain. Without accurate knowledge of ε, the knowledge of g also becomes erroneous; therefore, MGA yields a suboptimal $\langle H_u, x'_u \rangle$ for poisoning.

6 Potential Countermeasures

Having demonstrated that our BIAs are effective, we now turn to potential defenses and countermeasures, i.e., how can one defend against our BIAs? In this section, we propose three potential countermeasures and evaluate their effectiveness.

6.1 Noisy (Imperfect) Knowledge of f

Our BIAs with the informed adversary assume that the adversary has access to the true frequency distribution f. A natural question and potential countermeasure against informed BIAs can be: How would the BIAs perform if the adversary only had access to a noisy version of f instead? To investigate this, we perturbed the adversary's f with Gaussian noise (mean 0 and standard deviation σ), and evaluated the impact of this noise on the accuracy of our BIAs. We used both the MAE and MSE metrics to quantify the prediction error. We varied σ from 0.001 (small perturbation) to 0.1 (substantial perturbation), and measured the resulting errors under different real privacy budgets ε^*. Results for all five protocols (GRR, RAPPOR, OUE, OLH, and

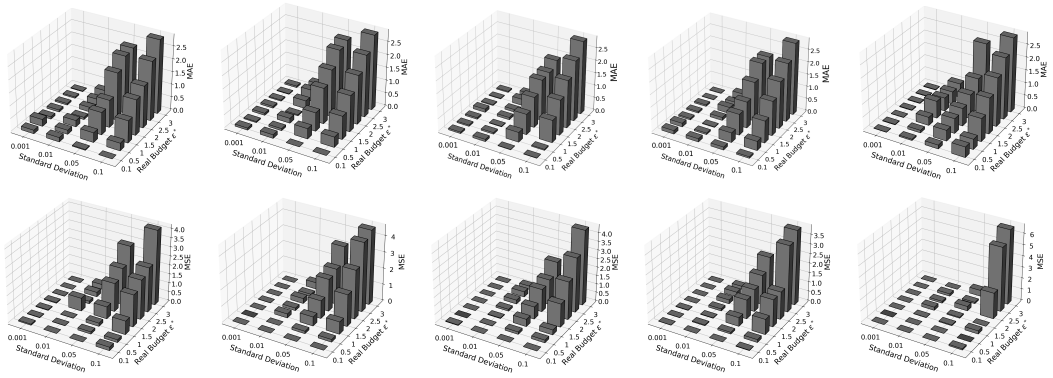


Fig. 9. Impact of Gaussian noise addition to f under varying degrees of Gaussian noise (controlled by the standard deviation σ) and real budget ε^*. Results are shown for GRR, RAPPOR, OUE, OLH, and SW protocols from left to right. First row uses MAE as the error metric, second row uses MSE as the error metric.

SW) are shown in Figure 9. The first row corresponds to MAE, and the second row corresponds to MSE.

The results demonstrate that both error metrics capture similar trends, indicating that limited noise can reduce the effectiveness of the attacks, while stronger noise leads to substantial prediction errors. For example, with $\sigma = 0.01$ and $\varepsilon^* = 2$ or 2.5, the MAE reaches approximately 0.5, corresponding to a 20–25% error in the predicted budgets. Although these errors do not eliminate the threat of BIAs, they show that highly accurate inference can be hindered with relatively small perturbations. For stronger protection (e.g., MAE or MSE approaching the value of ε^*), larger noise levels such as $\sigma = 0.05$ or 0.1 are required. However, such levels of noise can severely distort f, reducing its utility for legitimate downstream tasks.

6.2 Using Suboptimal Protocol Parameters

Our uninformed BIAs for some protocols (such as OLH and SW) make use of the protocols' internal parameters. For example, our BIA for OLH makes use of g, and our BIA for SW makes use of b. After the adversary estimates g and b from the observations $\mathcal{O}$, the adversary exploits the mathematical relationship between g and ε (or between b and ε) to infer ε. Then, one possible countermeasure could be to break these relationships by using suboptimal protocol parameters. That is, instead of using the recommended g and b values (which are mathematically related to ε), arbitrary values of g and b can be used to prevent the BIAs.

While we expect this to be an effective countermeasure, its utility impact also needs to be examined since the use of suboptimal protocol parameters will likely yield increased utility loss. To quantify utility loss, we define the ULoss metric as

$$\text{ULoss} = \frac{\sum_{v \in \mathcal{U}} |\tilde{f}_v - f_v|}{|\mathcal{U}|}. \tag{44}$$

In Figure 10, we execute this countermeasure on the OLH and SW protocols, and analyze its impacts in terms of MAE and ULoss. For OLH, we observe that in both plots (with $\varepsilon^* = 1.5$ and $\varepsilon^* = 2.5$) the MAEs of our BIAs are lowest when the protocol parameters are optimal. In addition, ULoss is also lowest when the protocol parameters are optimal. As the protocol parameters diverge from their optimal values, both MAEs and ULoss values start to increase. MAEs and ULoss values

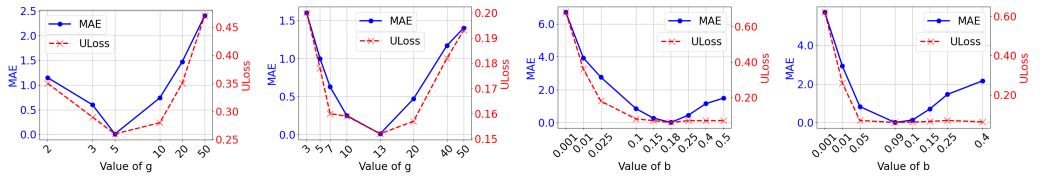


Fig. 10. MAE and ULoss impacts of using suboptimal g parameter in OLH (two graphs on the left), suboptimal b parameter in SW (two graphs on the right). Adult dataset is used in all experiments. For the OLH graphs on the left, first one is with $\varepsilon^* = 1.5$ (optimal g is 5), second one is with $\varepsilon^* = 2.5$ (optimal g is 13). For the SW graphs on the right, first one is with $\varepsilon^* = 1.5$ (optimal b is 0.18), second one is with $\varepsilon^* = 2.5$ (optimal b is 0.09).

generally seem to be correlated. Hence, we find that a defender who uses this countermeasure on the OLH protocol has to pay the price of increased ULoss.

In Figure 10, the two plots on the right use the SW protocol. Similar to the OLH protocol, MAEs and ULoss values are lowest in the SW protocol when the optimal value of b is used. As b values diverge from their optimal values, both MAEs and ULoss values start to increase. Yet, differently from OLH, we observe that although b values smaller than the optimal b cause substantial increase in ULoss, b values larger than the optimal b do not cause that much increase in ULoss. For example, when $\varepsilon^* = 1.5$, b values between 0.25 and 0.5 cause relatively small increases in ULoss compared to $b = 0.18$. On the other hand, non-negligible increases in MAE can be achieved in this range, e.g., MAE can increase from 0.02 when $b = 0.18$ to 1.49 when $b = 0.5$. Thus, in case of the SW protocol, the ULoss price that needs to be paid to increase MAE by a substantial amount can be relatively small. This example shows that good tradeoffs between MAE and ULoss may actually be found using this countermeasure in the SW protocol. Thus, we can recommend the use of suboptimal, larger b values in SW as a practical countermeasure against our BIAs for SW.

6.3 Using Suboptimal Perturbation Probabilities

Another potential countermeasure could be the use of suboptimal or randomized perturbation probabilities on the user side, i.e., at the perturbation stage. This way, the adversary's expectation (e.g., $\mathbb{E}[N'_u]$) will not match the observed protocol output, or it will cause the adversary to predict a different ε value (different from ε^*) because suboptimal probabilities cause the observed protocol output to better fit the expected results of a different ε value rather than the real ε^* value. On the other hand, a critical aspect of this countermeasure is to ensure that, despite the use of suboptimal probabilities, each user's perturbation satisfies ε^*-LDP for the desired ε^* budget. In other words, the use of suboptimal probabilities should not arbitrarily relax the privacy guarantees of the LDP protocol in such a way that LDP becomes violated when suboptimal probabilities are used.

Taking into account the above aspects, we implement this countermeasure on GRR and RAPPOR protocols as follows. Recall from Equation (2) that by default, in order to satisfy ε^*-LDP, GRR uses the following perturbation probabilities:

$$p = \frac{e^{\varepsilon^*}}{e^{\varepsilon^*} + |\mathcal{U}| - 1} \text{ and } q = \frac{1-p}{|\mathcal{U}| - 1} = \frac{1}{e^{\varepsilon^*} + |\mathcal{U}| - 1}. \tag{45}$$

This satisfies ε^*-LDP because $\frac{p}{q} = e^{\varepsilon^*}$. Here, if we increase p, this will naturally cause q to decrease, and this will yield a violation of ε^*-LDP because $\frac{p}{q} \leq e^{\varepsilon^*}$ will no longer hold. Instead, we implement suboptimality by decreasing p, which naturally causes q to increase. We implement the decrease in p by subtracting a random positive sample drawn from a Gaussian distribution $\mathcal{G}(0, \sigma)$ with mean 0 and standard deviation σ from the optimal value of p given in Equation (45). Then, we calculate the suboptimal (increased) q from the suboptimal (decreased) p using the equation: $q = \frac{1-p}{|\mathcal{U}|-1}$.

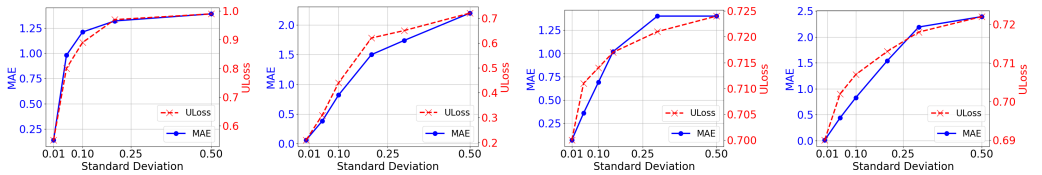


Fig. 11. MAE and ULoss impacts of using suboptimal perturbation probabilities in GRR (two graphs on the left) and RAPPOR (two graphs on the right). Adult dataset is used in all experiments. For the GRR graphs, first one is with $\varepsilon^* = 1.5$, second one is with $\varepsilon^* = 2.5$. For the RAPPOR graphs, first one is with $\varepsilon^* = 1.5$, second one is with $\varepsilon^* = 2.5$.

In RAPPOR, the LDP guarantees come from the bit keeping and bit flipping probabilities. In order to satisfy ε^*-LDP, RAPPOR uses the bit keeping probability $\alpha = \frac{e^{\varepsilon^*/2}}{e^{\varepsilon^*/2}+1}$ and, therefore, the bit flipping probability is $1 - \alpha = \frac{1}{e^{\varepsilon^*/2}+1}$. We should not increase α since increasing α may cause a violation of ε^*-LDP. Hence, we implement suboptimality in RAPPOR by decreasing α. Similar to GRR, we decrease α by subtracting a random positive sample drawn from a Gaussian distribution $\mathcal{G}(0, \sigma)$ with mean 0 and standard deviation σ from the optimal $\alpha = \frac{e^{\varepsilon^*/2}}{e^{\varepsilon^*/2}+1}$. The corresponding bit flipping probability is then calculated as $1 - \alpha$. Each user uses their suboptimal (randomized with Gaussian noise) bit keeping and bit flipping probabilities to perturb their bitvectors.

In both GRR and RAPPOR, the use of suboptimal perturbation probabilities will likely yield increased utility loss due to the additional noise introduced by suboptimality. Similar to the previous section, we utilize the ULoss metric to quantify this utility loss.

In Figure 11, we apply these countermeasures and analyze their impacts with respect to ULoss and MAE on the GRR and RAPPOR protocols, respectively. In both protocols, increasing the standard deviation σ of the Gaussian noise increases the suboptimality of perturbation. As a result, we observe that ULoss values in both protocols increase. In addition, the MAEs of BIAs also increase. As σ goes from 0.01 to 0.50, MAEs go from near-zero to MAE > 1 for $\varepsilon^* = 1.5$, and MAE > 2 for $\varepsilon^* = 2.5$. Such large MAEs relative to the ε^* budgets indicate that the BIAs are no longer effective.

Yet, the increasing ULoss amounts also need to be taken into account. For GRR, increases in ULoss are substantial, as ULoss goes from less than 0.6 to more than 0.9 when $\varepsilon^* = 1.5$, and it goes from 0.2 to 0.7 when $\varepsilon = 2.5$. Hence, for the GRR protocol, although the countermeasure is effective, there seems to be a large ULoss price that needs to be paid. Smaller σ such as $\sigma \in [0.01, 0.10]$ can be preferred in GRR so that MAE close to 0.5 or 1.0 can be achieved (which gives a reasonable amount of protection against our BIAs) while incurring moderate ULoss. For RAPPOR, although ULoss increases as σ increases, the amount of increase is much smaller. Thus, larger σ can be preferred in RAPPOR to achieve strong protection against BIAs while incurring moderate ULoss.

7 Related Work

Our work contributes to the recently growing literature on security and privacy attacks in LDP by proposing a new type of attack: BIA. Our BIAs are particularly relevant to IoT and edge analytics, due to the existence of streaming data, heterogeneous data collection architectures, and diverse data and user types with potentially different ε preferences. Below, we first survey recent works on applications of LDP on IoT and edge analytics. Then, we survey works on poisoning attacks and privacy attacks in LDP. While BIAs constitute a type of privacy attack, to the best of our knowledge, there does not exist any previous work with the goal of inferring privacy budgets in LDP. Hence, our BIAs are different and novel compared to related works in terms of the end goal. We provide a comparison between our work and the prior works in Table 7.

Table 7. Comparison between Prior Works and Our Work

Attack Category	Prior Work	Differences to Our Work
Poisoning	Cao et al. [7]	Different threat model. Poisoning attacks assume that a portion of users in $\lvert\mathcal{P}\rvert$ are malicious and they perform active manipulation. The goal is to decrease the accuracy of the statistics recovered by the data collector. In BIAs, users are honest. The adversary is passive (no active LDP protocol manipulation), and the goal is to infer the privacy budget ε.
	Cheu et al. [9]	
	Li et al. [32, 33]	
	Wu et al. [58]	
	Imola et al. [26]	
	Wang et al. [53]	
	Zheng et al. [64]	
Privacy	Murakami et al. [37]	Adversary aims to perform re-identification (linkage), not budget inference.
	Gadotti et al. [18]	Adversary aims to infer the membership of a user's true value in a pool.
	Gursoy et al. [21, 22]	Adversary aims to infer users' true values, not budgets.
	Arcolezi et al. [3]	Focuses on re-identification and true value inference specifically for multi-dimensional data, no budget inference.
	Chatzikokolakis et al. [8]	Evaluates protocols in terms of Bayes security, no budget inference.

7.1 Applications of LDP to IoT and Edge Analytics

LDP has seen widespread adoption in numerous fields related to the IoT and edge analytics, such as geolocation data [12, 24, 52], indoor positioning [30, 38], health data and wearables [31, 36, 57], Internet of Vehicles [63], cybersecurity [23], smart meters and smart grids [19, 41], as well as distributed deep learning [2, 49, 51]. In [61], Xu et al. proposed an LDP-based obfuscation framework for IoT data analytics to aggregate and distill IoT data at the edge. In [60], an edge computing framework with LDP was developed for mobile data analytics. Wu et al. [57] proposed protocols for locally private estimation of physical symptom frequency of diseases in the **Internet of Medical Things** (**IoMT**). A singular spectrum analysis-based LDP method has been proposed in [41] to prevent inference of household appliances in smart homes. In [19], a data aggregation scheme with LDP was developed based on randomized response. LDP was applied to smart grid data in [29] and smart home data in [50]. In [39], an LDP-based scheme was proposed for longitudinal and multidimensional range query answering in IoT. In [43], an optimization-based strategy was proposed for ε budget selection in smart city and smart home deployments of LDP.

Streaming data is prevalent in IoT and edge analytics, and there exist several LDP-based solutions that target data streams. Our BIAs are relevant to these scenarios since several works that target data streams perform uneven budget allocations, thereby using different budgets at different timestamps. Inference of these budgets can enable the adversary to learn information regarding the amount of change in the data stream, sensitivity of the stream, or the time-dependent utility requirements. Errounda and Liu [15] proposed a sliding window-based approach to release location statistics with LDP over multiple timestamps. Bao et al. [5] proposed CGM for enforcing (ε, δ)-LDP on streaming data collection using temporally correlated noise. In [42], Ren et al. proposed LDP-IDS, which utilizes the w-event privacy paradigm to provide privacy guarantees for infinite streams. Fu et al. [17] proposed a relaxed LDP variant called UDLDP for the collection of high-dimensional and correlated streaming sensor data. Gao and Zhou [20] used an adaptive budget allocation method to

perturb data points in time series and satisfy w-event privacy. Li et al. [34] studied the problem of top-k heavy hitter detection in data streams under bounded memory.

7.2 Poisoning Attacks in LDP

In poisoning attacks, malicious users poison their LDP protocol inputs or outputs to manipulate the statistics recovered on the data collector's side. The vulnerability of LDP protocols to poisoning was shown by Cheu et al. [9] and Cao et al. [7], who demonstrated their attacks using well-known LDP protocols (such as GRR, OUE, and OLH) and the frequency estimation task. Poisoning attacks for mean and variance estimation were proposed in [32]. Li et al. [33] studied the robustness of LDP protocols designed for numerical attributes against poisoning attacks. Wu et al. [58] proposed poisoning attacks to LDP protocols designed for key-value data collection. Imola et al. [26] studied the robustness of LDP protocols for graph analysis. Wang et al. [53] proposed methods for analyzing preference data under LDP with enhanced utility and robustness. Finally, Zheng et al. [64] studied data poisoning attacks against LDP-based privacy-preserving crowdsensing.

Defenses against poisoning attacks have also been proposed in the recent literature. Some preliminary defenses include adjusting the LDP protocol parameters, anomaly and outlier detection, as well as normalization and cleaning of poisoned server-side estimates [7, 9, 32, 58]. In addition, there exist several dedicated or stand-alone defense strategies. Kato et al. [28] developed verifiable LDP protocols to prevent manipulation attacks. They employed the **Cryptographic Randomized Response Technique** (**CRRT**) as a key component to construct verifiable versions of LDP mechanisms. Song et al. [47] analyzed attacks on GRR and OUE protocols, and proposed a defense strategy to enforce verifiability using an interactive framework where the user and the data collector complete the perturbation together. Huang et al. [25] introduced LDPGuard, which first estimates the percentage of malicious users and then provides adversarial schemes to defend against particular data poisoning attacks.

7.3 Privacy Attacks in LDP

Privacy attacks typically aim at inferring sensitive information from LDP protocol outputs or quantifying the practical privacy amounts achieved by LDP protocols against specific adversaries. Murakami et al. [37] studied the risk of re-identification in LDP protocols, in which an adversary tries to identify the user who sent a perturbed output. Gadotti et al. [18] proposed pool inference attacks in which an adversary tries to infer whether a user's true value falls within a pool of interest or which pool the user's true value falls into. Gursoy et al. [22] analyzed the privacy disclosure of LDP protocols when attacked by a Bayesian adversary that aims to infer users' true values from protocol outputs. In a later work, Gursoy et al. [21] used a similar Bayesian adversary formulation to propose longitudinal attacks on iterative data collection with LDP, to illustrate how collecting multiple responses from the same user can degrade that user's privacy. Arcolezi et al. [3] studied the risks stemming from multi-dimensional data collection with LDP, and proposed re-identification and attribute inference attacks targeting multi-dimensional data. Finally, the Bayes security measure proposed in [8] quantifies the expected gain of an adversary who observes the output of a perturbation mechanism. While the Bayes security measure is not specific to LDP, it was used to analyze the GRR protocol as a representative LDP mechanism.

8 Conclusion

In this article, we proposed a new class of attacks called BIAs on LDP data collection. We developed BIAs for two types of adversaries: informed and uninformed. We experimentally showed that our BIAs are effective, e.g., in more than 73% of the experiment settings, our BIAs enable the adversary to infer the ε budget with less than 0.03 MAE. We also proposed and evaluated three potential

countermeasures against our BIAs. Results show that while our countermeasures can reduce BIA accuracy, they also incur utility loss for the collected data. We analyzed the tradeoffs between utility loss and reduction in BIA accuracy, and provided recommendations regarding the practical uses of our countermeasures and their parameters.

Considering that BIAs constitute a new class of attacks, there are several avenues for future work. First, as we showed experimentally, BIA effectiveness may decrease when $|\mathcal{P}|$ is smaller, especially for informed adversaries. In response to this, in future work, we plan to develop BIAs that are effective under small $|\mathcal{P}|$. Second, we plan to evaluate our countermeasures more broadly, e.g., using more datasets, utility loss metrics, and ε^* budgets to better quantify their tradeoffs under various settings. Finally, based on the results of our analysis, we plan to propose countermeasures that yield better tradeoffs in specific settings (e.g., under targeted utility metrics).

References

[1] 2020. Apple—Learning with privacy at scale. Retrieved 8 January 2025 from https://docs-assets.developer.apple.com/ml-research/papers/learning-with-privacy-at-scale.pdf

[2] Pathum Chamikara Mahawaga Arachchige, Peter Bertok, Ibrahim Khalil, Dongxi Liu, Seyit Camtepe, and Mohammed Atiquzzaman. 2019. Local differential privacy for deep learning. *IEEE Internet of Things Journal* 7, 7 (2019), 5827–5842.

[3] Héber H. Arcolezi, Sébastien Gambs, Jean-François Couchot, and Catuscia Palamidessi. 2023. On the risks of collecting multidimensional data under local differential privacy. *Proceedings of the VLDB Endowment* 16, 5 (2023), 1126–1139.

[4] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2019. The privacy blanket of the shuffle model. In *Proceedings of the 39th Annual International Cryptology Conference*. Springer, 638–667.

[5] Ergute Bao, Yin Yang, Xiaokui Xiao, and Bolin Ding. 2021. CGM: An enhanced mechanism for streaming data collection with local differential privacy. *Proceedings of the VLDB Endowment* 14, 11 (2021), 2258–2270.

[6] Raef Bassily and Adam Smith. 2015. Local, private, efficient protocols for succinct histograms. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*. ACM, 127–135.

[7] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2021. Data poisoning attacks to local differential privacy protocols. In *Proceedings of the 30th USENIX Security Symposium*. 947–964.

[8] Konstantinos Chatzikokolakis, Giovanni Cherubin, Catuscia Palamidessi, and Carmela Troncoso. 2023. Bayes security: A not so average metric. In *Proceedings of the IEEE 36th Computer Security Foundations Symposium (CSF)*. IEEE, 388–406.

[9] Albert Cheu, Adam Smith, and Jonathan Ullman. 2021. Manipulation attacks in local differential privacy. In *Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 883–900.

[10] Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. 2018. Privacy at scale: Local differential privacy in practice. In *Proceedings of the 2018 International Conference on Management of Data*. ACM, 1655–1658.

[11] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting telemetry data privately. In *Proceedings of the Advances in Neural Information Processing Systems*. 3571–3580.

[12] Yuntao Du, Yujia Hu, Zhikun Zhang, Ziquan Fang, Lu Chen, Baihua Zheng, and Yunjun Gao. 2023. LDPTrace: Locally differentially private trajectory synthesis. *Proceedings of the VLDB Endowment* 16, 8 (2023), 1897–1909.

[13] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2468–2479.

[14] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1054–1067.

[15] F. Z. Errounda and Y. Liu. 2021. Collective location statistics release with local differential privacy. *Future Generation Computer Systems* 124 (2021), 174–186.

[16] Vitaly Feldman, Audra McMillan, and Kunal Talwar. 2022. Hiding among the clones: A simple and nearly optimal analysis of privacy amplification by shuffling. In *Proceedings of the IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, 954–964.

[17] Y. Fu, Q. Ye, R. Du, and H. Hu. 2023. Collecting multi-type and correlation-constrained streaming sensor data with local differential privacy. *ACM Transactions on Sensor Networks*.

[18] Andrea Gadotti, Florimond Houssiau, Meenatchi Sundaram Muthu Selva Annamalai, and Yves-Alexandre de Montjoye. 2022. Pool inference attacks on local differential privacy: Quantifying the privacy guarantees of apple's count mean sketch in practice. In *Proceedings of the 31st USENIX Security Symposium*. 501–518.

[19] Na Gai, Kaiping Xue, Bin Zhu, Jiayu Yang, Jianqing Liu, and Debiao He. 2022. An efficient data aggregation scheme with local differential privacy in smart grid. *Digital Communications and Networks* 8, 3 (2022), 333–342.
[20] W. Gao and S. Zhou. 2023. Privacy-preserving for dynamic real-time published data streams based on local differential privacy. *IEEE Internet of Things Journal* 11, 8 (2023), 13551–13562.
[21] Mehmet Emre Gursoy. 2024. Longitudinal attacks against iterative data collection with local differential privacy. *Turkish Journal of Electrical Engineering and Computer Sciences* 32, 1 (2024), 198–218.
[22] M. E. Gursoy, L. Liu, K. H. Chow, S. Truex, and W. Wei. 2022. An adversarial approach to protocol analysis and selection in local differential privacy. *IEEE Transactions on Information Forensics and Security* 17 (2022), 1785–1799.
[23] Mehmet Emre Gursoy, Acar Tamersoy, Stacey Truex, Wenqi Wei, and Ling Liu. 2019. Secure and utility-aware data collection with condensed local differential privacy. *IEEE Transactions on Dependable and Secure Computing* 18, 5 (2019), 2365–2378.
[24] D. Hong, W. Jung, and K. Shim. 2022. Collecting geospatial data under local differential privacy with improving frequency estimation. *IEEE Transactions on Knowledge and Data Engineering* 35, 7 (2022), 6739–6751.
[25] K. Huang, G. Ouyang, Q. Ye, H. Hu, B. Zheng, X. Zhao, R. Zhang, and X. Zhou. 2024. LDPGuard: Defenses against data poisoning attacks to local differential privacy protocols. *IEEE Transactions on Knowledge and Data Engineering* 36, 7 (2024), 3195–3209.
[26] Jacob Imola, Amrita Roy Chowdhury, and Kamalika Chaudhuri. 2022. Robustness of locally differentially private graph analysis against poisoning. arXiv:2210.14376. Retrieved from https://arxiv.org/abs/2210.14376
[27] Bargav Jayaraman and David Evans. 2022. Are attribute inference attacks just imputation?. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 1569–1582.
[28] Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. 2021. Preventing manipulation attack in local differential privacy using verifiable randomization mechanism. In *Proceedings of the 35th Annual IFIP WG 11.3 Conference on Data and Applications Security and Privacy (DBSec)*. Springer, 43–60.
[29] Zeynep Sila Kaya and Mehmet Emre Gursoy. 2023. On the effectiveness of re-identification attacks and local differential privacy-based solutions for smart meter data. In *Proceedings of the SECRYPT*. 111–122.
[30] Jong Wook Kim and Beakcheol Jang. 2019. Workload-aware indoor positioning data collection via local differential privacy. *IEEE Communications Letters* 23, 8 (2019), 1352–1356.
[31] Jong Wook Kim, Jong Hyun Lim, Su Mee Moon, and Beakcheol Jang. 2019. Collecting health lifelog data from smartwatch users in a privacy-preserving manner. *IEEE Transactions on Consumer Electronics* 65, 3 (2019), 369–378.
[32] Xiaoguang Li, Ninghui Li, Wenhai Sun, Neil Zhenqiang Gong, and Hui Li. 2023. Fine-grained poisoning attack to local differential privacy protocols for mean and variance estimation. In *Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23)*. 1739–1756.
[33] X. Li, Z. Li, N. Li, and W. Sun. 2025. On the robustness of ldp protocols for numerical attributes under data poisoning attacks. In *Proceedings of the Network and Distributed System Security (NDSS) Symposium 2025*.
[34] Xiaochen Li, Weiran Liu, Jian Lou, Yuan Hong, Lei Zhang, Zhan Qin, and Kui Ren. 2024. Local differentially private heavy hitter detection in data streams with bounded memory. *Proceedings of the ACM on Management of Data* 2, 1 (2024), 1–27.
[35] Zitao Li, Tianhao Wang, Milan Lopuhaä-Zwakenberg, Ninghui Li, and Boris Škoric. 2020. Estimating numerical distributions under local differential privacy. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 621–635.
[36] Thomas Marchioro, Andrei Kazlouski, and Evangelos P Markatos. 2023. Practical crowdsourcing of wearable IoT data with local differential privacy. In *Proceedings of the 8th ACM/IEEE Conference on Internet of Things Design and Implementation*. 275–287.
[37] Takao Murakami and Kenta Takahashi. 2021. Toward evaluating re-identification risks in the local privacy model. *Transactions on Data Privacy* 14, 3 (2021), 79–116.
[38] Hojjat Navidan, Vahideh Moghtadaiee, Niki Nazaran, and Mina Alishahi. 2022. Hide me behind the noise: Local differential privacy for indoor location privacy. In *Proceedings of the 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 514–523.
[39] Y. Ni, J. Li, W. Chang, and J. Xiao. 2023. An LDP-Based Privacy-Preserving Longitudinal and Multidimensional Range Query Scheme in IoT. *IEEE Internet of Things Journal* 11, 3 (2023), 5210–5221.
[40] Yiwen Nie, Wei Yang, Liusheng Huang, Xike Xie, Zhenhua Zhao, and Shaowei Wang. 2019. A utility-optimized framework for personalized private histogram estimation. *IEEE Transactions on Knowledge and Data Engineering* 31, 04 (2019), 655–669.
[41] Lu Ou, Zheng Qin, Shaolin Liao, Tao Li, and Dafang Zhang. 2020. Singular spectrum analysis for local differential privacy of classifications in the smart grid. *IEEE Internet of Things Journal* 7, 6 (2020), 5246–5255.
[42] Xuebin Ren, Liang Shi, Weiren Yu, Shusen Yang, Cong Zhao, and Zongben Xu. 2022. LDP-IDS: Local differential privacy for infinite data streams. In *Proceedings of the 2022 International Conference on Management of Data*. 1064–1077.

[43] S. Seyedkazemi, M. E. Gursoy, and Y. Saygin. 2024. Capacity planning under local differential privacy with optimized budget selection. *IEEE Transactions on Industrial Informatics*.
[44] Zixuan Shen, Zhihua Xia, and Peipeng Yu. 2021. Pldp: Personalized local differential privacy for multidimensional data aggregation. *Security and Communication Networks* 2021, 1 (2021), 6684179.
[45] Mary Anne Smart, Dhruv Sood, and Kristen Vaccaro. 2022. Understanding risks of privacy theater with differential privacy. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW2 (2022), 1–24.
[46] H. Song, H. Shen, N. Zhao, Z. He, M. Wu, W. Xiong, and M. Zhang. 2024. APLDP: Adaptive personalized local differential privacy data collection in mobile crowdsensing. *Computers & Security* 136 (2024), 103517.
[47] S. Song, L. Xu, and L. Zhu. 2023. Efficient defenses against output poisoning attacks on local differential privacy. *IEEE Transactions on Information Forensics and Security* 18 (2023), 5506–5521.
[48] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. 2017. Privacy loss in Apple's implementation of differential privacy on macOS 10.12. arXiv:1709.02753. Retrieved from https://arxiv.org/abs/1709.02753
[49] Stacey Truex, Ling Liu, Ka-Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. 2020. LDP-Fed: Federated learning with local differential privacy. In *Proceedings of the 3rd ACM International Workshop on Edge Systems, Analytics and Networking*. 61–66.
[50] Nazar Waheed, Fazlullah Khan, Spyridon Mastorakis, Mian Ahmad Jan, Abeer Z. Alalmaie, and Priyadarsi Nanda. 2023. Privacy-enhanced living: A local differential privacy approach to secure smart home data. In *Proceedings of the 2023 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*. IEEE, 1–6.
[51] Baocang Wang, Yange Chen, Hang Jiang, and Zhen Zhao. 2023. Ppefl: Privacy-preserving edge federated learning with local differential privacy. *IEEE Internet of Things Journal* 10, 17 (2023), 15488–15500.
[52] Han Wang, Hanbin Hong, Li Xiong, Zhan Qin, and Yuan Hong. 2022. L-SRR: Local differential privacy for location-based services with staircase randomized response. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. Association for Computing Machinery, New York, NY, USA, 2809–2823.
[53] Shaowei Wang, Xuandi Luo, Yuqiu Qian, Jiachun Du, Wenqing Lin, and Wei Yang. 2022. Analyzing preference data with local privacy: Optimal utility and enhanced robustness. *IEEE Transactions on Knowledge and Data Engineering* 35, 8 (2022), 7753–7767.
[54] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally differentially private protocols for frequency estimation. In *Proceedings of the 26th USENIX Security Symposium*. 729–745.
[55] Tianhao Wang, Ninghui Li, and Somesh Jha. 2018. Locally differentially private frequent itemset mining. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*. IEEE.
[56] Stanley L. Warner. 1965. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association* 60, 309 (1965), 63–69.
[57] X. Wu, M. R. Khosravi, L. Qi, G. Ji, W. Dou, and X. Xu. 2020. Locally private frequency estimation of physical symptoms for infectious disease analysis in Internet of Medical Things. *Computer Communications* 162 (2020), 139–151.
[58] Yongji Wu, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2022. Poisoning attacks to local differential privacy protocols for key-value data. In *Proceedings of the 31st USENIX Security Symposium*. 519–536.
[59] X. Xiong, S. Liu, D. Li, Z. Cai, and X. Niu. 2020. A comprehensive survey on local differential privacy. *Security and Communication Networks* 2020, 1 (2020), 8829523.
[60] Chugui Xu, Ju Ren, Liang She, Yaoxue Zhang, Zhan Qin, and Kui Ren. 2019. EdgeSanitizer: Locally differentially private deep inference at the edge for mobile data analytics. *IEEE Internet of Things Journal* 6, 3 (2019), 5140–5151.
[61] Chugui Xu, Ju Ren, Deyu Zhang, and Yaoxue Zhang. 2018. Distilling at the edge: A local differential privacy obfuscation framework for IoT data analytics. *IEEE Communications Magazine* 56, 8 (2018), 20–25.
[62] M. Yang, T. Guo, T. Zhu, I. Tjuawinata, J. Zhao, and K. Y. Lam. 2024. Local differential privacy and its applications: A comprehensive survey. *Computer Standards & Interfaces* 89 (2024), 103827.
[63] Ping Zhao, Guanglin Zhang, Shaohua Wan, Gaoyang Liu, and Tariq Umer. 2020. A survey of local differential privacy for securing internet of vehicles. *The Journal of Supercomputing* 76, 11 (2020), 8391–8412.
[64] Z. Zheng, Z. Li, C. Huang, S. Long, M. Li, and X. Shen. 2024. Data poisoning attacks and defenses to LDP-based privacy-preserving crowdsensing. *IEEE Transactions on Dependable and Secure Computing* 21, 5 (2024), 4861–4878.

Appendices

A Numerical Examples for Theoretical Analysis

Consider a setup with $|\mathcal{U}| = 74$ and $|\mathcal{P}| = 45222$. Suppose the true privacy budget is $\varepsilon_{\text{true}} = 1.0$, and the initial search interval is $[0.1, 5]$ with $\tau = 0.001$. According to GRR's theoretical analysis, $\sigma \approx \sqrt{\frac{|\mathcal{U}|}{|\mathcal{P}|}} \approx 0.0404$, and $m = \lceil \log_2(4.9/0.001) \rceil = 13$. Estimating ρ_j for the first few steps and

summing contributions, we find: $\sum_{j=1}^{13} \rho_j \cdot \frac{R_{j-1}}{2} \approx 0.1613$. This leads to a final expected error of:

$$\mathbb{E}[|\hat{\varepsilon} - \varepsilon_{\text{true}}|] \approx 0.1613 + 0.00025 = 0.1616 \tag{46}$$

which shows direct correlation with the results we present in Table 1.

Using the same setup but this time performing the computation for RAPPOR, we find: $\alpha = 0.6225$, $\sigma = 0.0806$, and $m = 13$. Evaluating the sum for ρ_j gives: $\sum_{j=1}^{13} \rho_j \frac{R_{j-1}}{2} \approx 0.0798$. This leads to a final expected error of

$$\mathbb{E}\left[|\hat{\varepsilon} - \varepsilon_{\text{true}}|\right] \approx 0.0798 + 0.00025 = 0.0801. \tag{47}$$

This theoretical estimate closely matches (though slightly overshoots) the experimental error in Table 1. The remaining gap can be explained by LDP randomness and/or effects beyond the Gaussian approximation we use in our analysis.

B Attack Algorithm for Uninformed BIA on GRR

The pseudocode for the attack is provided in Algorithm 2.

ALGORITHM 2: Uninformed BIA for GRR

Input : $\mathcal{S}, \mathcal{O}, \varepsilon_{\min}, \varepsilon_{\max}, \tau$
Output: Predicted ε

$\tilde{\mathcal{S}} \leftarrow \sum_{y_u \in \mathcal{O}} y_u$
$target \leftarrow |\mathcal{S} - \tilde{\mathcal{S}}|$
$\mathcal{P}_{syn} \leftarrow$ Generate synthetic user population with values $\{v_1^s, v_2^s, \ldots, v_{|\mathcal{P}_{syn}|}^s\}$ such that $\mathcal{S} = \sum_{i=1}^{|\mathcal{P}_{syn}|} v_i^s$
Initialize $\varepsilon_{\text{low}} \leftarrow \varepsilon_{\min}$ and $\varepsilon_{\text{high}} \leftarrow \varepsilon_{\max}$
while $\varepsilon_{high} - \varepsilon_{low} > \tau$ **do**
    $\varepsilon_{\text{mid}} \leftarrow \frac{\varepsilon_{\text{high}} + \varepsilon_{\text{low}}}{2}$
    $\mathcal{O}_{syn} \leftarrow$ Simulate LDP data collection using the GRR protocol with $\mathcal{P}_{syn}$ and budget ε_{mid}
    $\mathcal{S}_{syn} \leftarrow \sum_{y \in \mathcal{O}} y$
    $current \leftarrow |\mathcal{S} - \mathcal{S}_{syn}|$
    if $current < target$ **then**
        $\varepsilon_{\text{high}} \leftarrow \varepsilon_{\text{mid}}$
    else
        $\varepsilon_{\text{low}} \leftarrow \varepsilon_{\text{mid}}$
return ε_{low}

Received 12 January 2025; revised 1 June 2025; accepted 29 July 2025