GPM: The Gaussian Pancake Mechanism for Planting Undetectable Backdoors in Differential Privacy

HAOCHEN SUN, University of Waterloo, Canada
XI HE, University of Waterloo, Canada

Differential privacy (DP) has become the gold standard for preserving individual privacy in data analysis. However, an implicit yet fundamental assumption underlying these rigorous privacy guarantees is the correct implementation and execution of DP mechanisms. Several incidents of unintended privacy loss have occurred due to numerical issues and inappropriate configurations of DP software, which have been successfully exploited in privacy attacks. To better understand the seriousness of defective DP software, we ask the following question: is it possible to elevate these passive defects into active privacy attacks while maintaining covertness?

To address this question, we present the *Gaussian pancake mechanism (GPM)*, a novel mechanism that is computationally indistinguishable from the widely used Gaussian mechanism (GM), yet exhibits arbitrarily weaker statistical DP guarantees. This unprecedented separation enables a new class of backdoor attacks: by indistinguishably passing off as the authentic GM, GPM can covertly degrade statistical privacy. Unlike the unintentional privacy loss caused by GM's numerical issues, GPM is an adversarial yet undetectable backdoor attack against data privacy. We formally prove GPM's covertness, characterize its statistical leakage, and demonstrate a concrete distinguishing attack that can achieve near-perfect success rates under suitable parameter choices, both theoretically and empirically.

Our results underscore the importance of using transparent, open-source DP libraries and highlight the need for rigorous scrutiny and formal verification of DP implementations to prevent subtle, undetectable privacy compromises in real-world systems.

CCS Concepts: • **Security and privacy → Data anonymization and sanitization**; **Information accountability and usage control**; **Malware and its mitigation**.

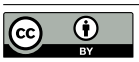
Additional Key Words and Phrases: Differential Privacy, Privacy Attack

ACM Reference Format:
Haochen Sun and Xi He. 2026. GPM: The Gaussian Pancake Mechanism for Planting Undetectable Backdoors in Differential Privacy. *Proc. ACM Manag. Data* 4, 3 (SIGMOD), Article 191 (June 2026), 25 pages. https://doi.org/10.1145/3802068

1 Introduction

In the modern digital landscape, an unprecedented volume of data is continuously collected, aggregated, and analyzed across diverse domains, including healthcare, finance, social networks, and cloud-based services. As data-driven technologies proliferate, ensuring the privacy of individuals represented in these datasets has become a critical and widely acknowledged challenge. Differential privacy (DP) [15, 16] has emerged as the de facto gold standard for formal privacy guarantees. It offers strong theoretical protection by ensuring that the presence or absence of a single individual's data has a provably bounded influence on the output of a computation, thereby statistically limiting the risk of privacy breaches against any adversary.

Authors' Contact Information: Haochen Sun, University of Waterloo, Waterloo, Canada, haochen.sun@uwaterloo.ca; Xi He, University of Waterloo, Waterloo, Canada, xi.he@uwaterloo.ca.



ACM 2836-6573/2026/6-ART191
https://doi.org/10.1145/3802068

However, realizing the strong theoretical guarantees offered by differential privacy in practice depends on a crucial yet often overlooked assumption: that the mechanisms are correctly implemented and faithfully executed. In other words, the software systems delivering differential privacy must precisely adhere to the formal design of the mechanisms. Any deviation, whether due to programming errors, numerical instability, or misinterpretation of the specification, can undermine the intended protections and leave users vulnerable to privacy breaches.

Indeed, several incidents have demonstrated the risks posed by compromised or improperly implemented differential privacy mechanisms. In particular, Laplace, Exponential, and Gaussian mechanisms have all been found vulnerable to numerical issues [28, 29, 39]. This is because the original definitions and guarantees of differential privacy are formulated in the real number space but must ultimately be realized using finite-precision floating-point arithmetic. Such discrepancies have opened the door to distinguishing attacks that exploit these vulnerabilities. In particular, by checking only the mechanism output, the attacker identifies the correct input database from a pair of neighbouring databases with a success rate of over 92% on widely used algorithms such as DP-SGD [29]. These findings have prompted the release of security patches in major differential privacy libraries [26]. Beyond numerical issues, practical misconfigurations have also raised concerns. For example, in 2017, it was revealed that earlier versions of macOS employed local differential privacy for data collection but used excessively large privacy budgets, rendering the guarantees effectively meaningless [56].

The existence of compromised DP software raises two natural and urgent questions. First, given that distinguishing attacks can exploit passive implementation issues, can such compromised implementations be intentionally elevated into *active* privacy attacks that an adversary engineers and uses? Second, how *covert* can these deviations be? In particular, intellectual-property concerns may keep deployed DP implementations closed-source. However, the aforementioned vulnerabilities resulting from numerical issues and excessive privacy budgets are still detectable by examining only the output distribution. But do such deviations exist, whether introduced deliberately or accidentally, that make the outputs computationally indistinguishable from the genuine outputs while leaving no obvious forensic traces (e.g., unexpected system calls and extra network traffic)?

If such deviations exist, they can be weaponized as backdoors, which substantially and silently undermine the formal privacy guarantees, either theoretically or empirically. A backdoor attacker could embed a compromised implementation into a closed-source DP library or service so that, to all ordinary examinations and black-box tests, or to the server executing the mechanisms itself, the mechanism appears correct and privacy-preserving. Unless a full white-box inspection of the implementation is performed (which is infeasible for many proprietary systems), neither servers that rely on the library nor downstream users can detect the backdoor or realize that the stated DP guarantees have been violated. The backdoor would therefore elevate from accidental and passive leakage (e.g., rounding or floating-point artifacts) to intentional, adversarial insertion of covert vulnerabilities that remain hidden to standard testing and auditing practices.

To investigate this threat, we introduce the **Gaussian pancake mechanism (GPM)**, an indistinguishable backdoor attack against data privacy, parameterized by a secret backdoor key. To any party not knowing the backdoor key, GPM is computationally indistinguishable from the widely used Gaussian mechanism (GM). That is, no polynomial-time adversary (including the server and the downstream users) can efficiently tell their outputs apart and thus discover the existence of the backdoor. Critically, however, under certain parameter regimes, GPM exhibits substantially weaker privacy guarantees: to the backdoor attacker knowing the secret backdoor key, arbitrarily large privacy leakage could be induced. Given that GM is ubiquitous in deployed systems, the existence of constructions like GPM leads to a troubling conclusion: **any non-open-source DP software is not safe for production use, even with auditing in place**. In particular, passing black-box

audits does not rule out the existence of undetectable backdoors in the software that violate the claimed privacy guarantees.

The existence of GPM has significant implications for the design and deployment of privacy-preserving systems. Most importantly, it highlights the necessity of using transparent, open-source implementations such as OpenDP [54], Opacus [61], and Diffprivlib [25]. Without access to and scrutiny of implementation details, backdoors like GPM could be seamlessly integrated and remain undetected, while appearing indistinguishable from authentic mechanisms. Moreover, further verification measures may be adopted to mitigate this risk. These include programming-language-based formal verification of DP libraries [35, 41, 50, 59], which can ensure semantic correctness from specification to implementation. Additionally, cryptographic techniques such as zero-knowledge proofs and verifiable computation [5, 31, 32, 40, 53, 55] can be used to verify that a mechanism was executed correctly without requiring trust in the server, which is particularly relevant in modern contexts as DP is increasingly adopted in security-critical applications.

Our contributions can be summarized as follows:

- We formally define the Gaussian pancake mechanism (GPM) and describe an indistinguishable backdoor attack based on this construction. **(Section 3)**
- We analyze the statistical privacy guarantees of GPM and show that, despite satisfying the same computational DP guarantees as the Gaussian mechanism (GM), GPM may require an arbitrarily large lower bound on ε to satisfy $(\varepsilon, \delta = 0.5)$-DP[1], rendering it unsuitable for practical deployment. **(Section 4)**
- We study the distinguishing attack, a concrete attack that exploits the additional privacy leakage introduced by the GPM backdoor, prove that its success rate can be arbitrarily close to 1, and discuss the potential mitigations against it. **(Sections 5 and 6)**
- We provide experimental evaluations of the GPM backdoor's effectiveness, demonstrating that distinguishing attacks exploiting GPM can achieve near-perfect success rates under suitable parameter choices. **(Section 7)**

2 Preliminaries

2.1 Assumptions and Notations

In this study, we assume that all involved parties, including servers, downstream users, supply-chain backdoor planters, backdoor privacy attackers, regular privacy attackers, and generic adversaries, are probabilistic polynomial-time (PPT). We also assume that all mechanisms discussed in this study are computable in polynomial time. The notations used throughout the paper are summarized in Table 1.

2.2 Privacy Notions

We first recall *differential privacy (DP)* (Definition 2.1), which is based on the statistical difference of mechanism outputs on neighbouring input databases.

Definition 2.1 (differential privacy [15, 16]). *Given $\varepsilon > 0$ and $0 \leq \delta < 1$, a mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{Y}$ is (ε, δ)-differentially private (DP) if, for any pair of neighbouring databases $D \simeq D'$ that differ in a single row, and any measurable subset $S \subseteq \mathcal{Y}$,*

$$\Pr\left[M(D) \in S\right] \leq e^{\varepsilon} \Pr\left[\mathcal{M}(D') \in S\right] + \delta. \tag{1}$$

[1] ε increases as δ decreases.

Table 1. Notations

Notation	Definition
$D \simeq D'$	D and D' are neighbouring databases
$\|\cdot\|$	the ℓ^2 norm in Euclidean space
$y \leftarrow\$ S$	y is independently sampled from the uniform distribution over a set S
$y \leftarrow\$ \mathcal{P}$	y is independently sampled from probability distribution $\mathcal{P}$
$y \leftarrow\$ \mathcal{F}(x)$	y is independently sampled from the output distribution of a randomized function $\mathcal{F}(x)$
$Y \sim S/\mathcal{P}/\mathcal{F}(x)$	a random variable Y follows the probability distribution as defined above
$\mathcal{P}_1 \equiv \mathcal{P}_2$	two probability distributions $\mathcal{P}_1$ and $\mathcal{P}_2$ are equivalent
$\mathsf{negl}(\kappa)$	the class of functions that are asymptotically smaller than any inverse polynomial, i.e., $\left\{f : 0 \le f(\kappa) < \kappa^{-O(1)}\right\}$
n	sample size
d	dimensionality of mechanism output

2.3 Gaussian Mechanism

The Gaussian mechanism (GM) [16] is one of the most prevalent DP mechanisms. We state its definition and provide its DP guarantee in Definition 2.2 and Theorem 2.3, respectively.

Definition 2.2 (Gaussian mechanism). *Given a query $q : \mathcal{D} \to \mathbb{R}^d$, the* Gaussian mechanism (GM) *is defined as the d-dimensional multivariate Gaussian distribution with mean $q(D)$ and covariance $\sigma^2 I_d$, i.e.,*

$$\mathcal{M}^*_\sigma(D) = \mathcal{N}\left(q(D), \sigma^2 I_d\right), \tag{2}$$

where $\sigma > 0$ and I_d is the d-dimensional identity matrix. We denote the probability density function (p.d.f.) of $\mathcal{M}^*_\sigma(D)$ as $f^*_\sigma(D)$.

Theorem 2.3. *For any $0 < \delta \le 0.5$ such that $\varepsilon = \frac{\Delta^2}{2\sigma^2} - \frac{\Delta}{\sigma}\Phi^{-1}(\delta)$, where $\Delta := \sup_{D \simeq D'} \|q(D) - q(D')\|$ is the sensitivity of query q, and Φ is the cumulative distribution function (c.d.f.) of the standard Gaussian distribution, $\mathcal{M}^*_\sigma(\cdot)$ is (ε, δ)-DP [16].*

Note that, compared with Theorem 2.3, a more commonly used yet looser closed-form upper bound for ε is $\frac{\Delta}{\sigma}\sqrt{2\ln\frac{1.25}{\delta}}$ for $0 < \varepsilon, \delta < 1$. However, due to the extreme values of ε involved in our analysis, we do not restrict ourselves to $\varepsilon < 1$ in this study.

2.3.1 Numerical Issues and Distinguishing Attacks. Despite the theoretical guarantees, actual floating-point implementations of GM have been found to result in enlarged query sensitivities [9, 39], and distinguishing attacks have been shown effective in exploiting this numerical issue. In particular, given a pair of neighbouring databases $D \simeq D'$ (with a randomly chosen record replaced) and a query result perturbed by Gaussian noise, the attacker's goal is to determine whether the result was computed on D or D' [29]. The existence of such attacks has necessitated security patches in DP libraries by further obfuscating the least significant bits [26].

2.3.2 Discrete Gaussian Mechanism (DGM). For queries with discrete outputs, one possible mitigation against the numerical issues of GM is the discrete Gaussian mechanism.

Definition 2.4 (discrete Gaussian mechanism [8]). *Given a query $q : \mathcal{D} \to \mathbb{Z}$, the* discrete Gaussian mechanism (DGM) *is defined as the discrete Gaussian distribution*

$$\mathcal{M}^{\mathbb{Z}}_\sigma(D) := \mathcal{N}_{\mathbb{Z}}\left(q(D), \sigma^2\right) \tag{3}$$

with probability mass function (p.m.f.)

$$\phi_{\mathbb{Z}}\left(z; q\left(D\right), \sigma^2\right) \propto \exp\left(-\frac{\left(z - q\left(D\right)\right)^2}{2\sigma^2}\right) \tag{4}$$

for any $z \in \mathbb{Z}$, with location $q(D)$ and scale parameter σ.

DGM has various applications, especially in distributed data analytics [30]. Nevertheless, the use of floating-point GM remains ubiquitous, particularly for queries with non-discrete outputs such as in DP-SGD [1].

2.4 Gaussian Pancake Distribution

The *Gaussian pancake distribution*, also known as the *continuous learning-with-error (CLWE) distribution* [7, 21], is the continuous analogue of the *learning-with-error (LWE)* distribution [36, 48]. While the LWE distribution has been widely used for encryption algorithms [18, 42, 43, 49], CLWE has primarily been applied in learning theory [11–13], especially in state-of-the-art diffusion models [10, 52]. Notably, in 2022, it was discovered that indistinguishable backdoors exist for certain deep neural network structures, and one of the constructions relies on the CLWE distribution [19]. In this study, it suffices to focus on a special case of the CLWE distribution, namely the homogeneous CLWE (hCLWE) distribution, introduced in Definition 2.5.

Definition 2.5 (hCLWE distribution [7]). *For any shape parameter $s > 0$ and d-dimensional real vector $\mathbf{x} \in \mathbb{R}^d$, let $\rho_s(\mathbf{x}) := \exp\left(-\pi\left\|\frac{\mathbf{x}}{s}\right\|^2\right)$, so that $\frac{\rho_s(\mathbf{x})}{s^d}$ is the p.d.f. of the d-dimensional multivariate Gaussian distribution with covariance $\frac{s^2}{2\pi} I_d$. The* homogeneous CLWE distribution *$\mathcal{H}_{\mathbf{w},\beta,\gamma}$ is parameterized by $\mathbf{w} \in \mathbb{S}^{d-1} := \left\{\mathbf{y} \in \mathbb{R}^d : \|\mathbf{y}\| = 1\right\}$ and $\beta, \gamma > 0$, such that the p.d.f. $\psi_{\mathbf{w},\beta,\gamma} : \mathbb{R}^d \to \mathbb{R}_+$ is defined for any $\mathbf{y} \in \mathbb{R}^d$ as*

$$\psi_{\mathbf{w},\beta,\gamma}(\mathbf{y}) \propto \rho_1(\mathbf{y}) \cdot \sum_{z \in \mathbb{Z}} \rho_\beta\left(z - \gamma \mathbf{y}^\top \mathbf{w}\right). \tag{5}$$

It is also worth noting that $\mathcal{H}_{\mathbf{w},\beta,\gamma}$ is equivalent to a mixture of Gaussians with equally spaced means and identical variance, as stated in Lemma 2.6 and illustrated in Figure 1. In particular, its Gaussian components are equally spaced along the direction of $\mathbf{w}$, with the distance between neighbouring components given by $\Theta\left(\frac{1}{\gamma}\right)$. Each Gaussian component has a width of $\Theta\left(\frac{\beta}{\gamma}\right)$ along $\mathbf{w}$, which is significantly smaller than the $\Theta(1)$ width in the orthogonal directions. Consequently, the probability density of hCLWE resembles pancakes stacked along the direction of $\mathbf{w}$, giving rise to the alias *Gaussian pancake distribution.*

Lemma 2.6. *For any $\mathbf{w} \in \mathbb{S}^{d-1}$ and $\beta, \gamma > 0$, the p.d.f. of the hCLWE distribution $\mathcal{H}_{\mathbf{w},\beta,\gamma}$ (as defined in Definition 2.5) can be expressed as*

$$\psi_{\mathbf{w},\beta,\gamma}(\mathbf{y}) \propto \sum_{z \in \mathbb{Z}} a_{\beta,\gamma,z} \cdot \phi\left(\mathbf{y}; \frac{\gamma z}{\beta^2 + \gamma^2}\mathbf{w}, \Sigma_{\mathbf{w},\beta,\gamma}\right) \tag{6}$$

where $a_{\beta,\gamma,z} := \exp\left(-\frac{\pi z^2}{\beta^2+\gamma^2}\right)$ and $\Sigma_{\mathbf{w},\beta,\gamma} := \frac{1}{2\pi}\left(I - \frac{\gamma^2}{\beta^2+\gamma^2}\mathbf{w}\mathbf{w}^\top\right)$, such that $\phi\left(\cdot; \frac{\gamma z}{\beta^2+\gamma^2}\mathbf{w}, \Sigma_{\mathbf{w},\beta,\gamma}\right)$ is the p.d.f. of the multivariate Gaussian distribution $\mathcal{N}\left(\frac{\gamma z}{\beta^2+\gamma^2}\mathbf{w}, \Sigma_{\mathbf{w},\beta,\gamma}\right)$ [7].

The hCLWE distribution and the multivariate Gaussian distribution are considered **computationally indistinguishable by polynomial-time sampling**, under the hardness assumption that at least one of the *Shortest Independent Vector Problem (SIVP)* or the *Gap Shortest Vector Problem (GapSVP)* is not solvable in bounded-error quantum polynomial time (BQP). Specifically,

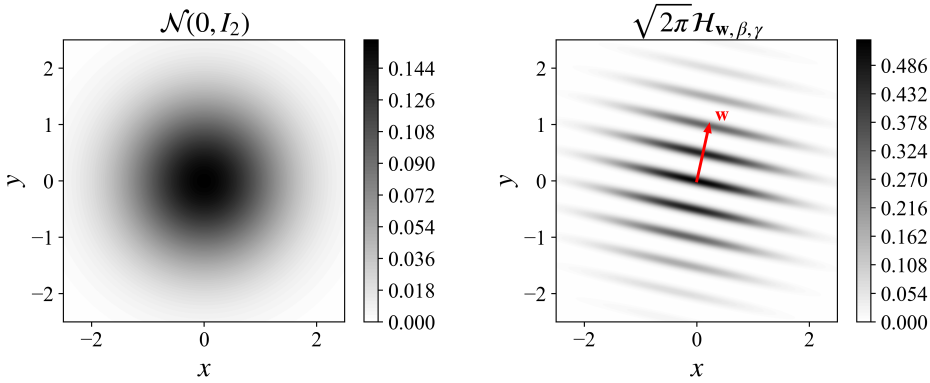


Fig. 1. Probability densities of the Gaussian distribution and the hCLWE distribution in $d = 2$. Note that distinguishing between the two distributions is assumed to be a hard problem for higher d, where d can be viewed as an instantiation of the security parameter κ.

given dimensionality d, an adversary is tasked with distinguishing between the d-dimensional hCLWE distribution and the d-dimensional multivariate Gaussian distribution, with a sample $s = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{n(d)})$ of sample size $n(d)$ polynomially large in d, and without knowing the value of $\mathbf{w}$. Theorem 2.7 states that, under the aforementioned hardness assumptions, no such PPT adversary exists.

Theorem 2.7. *Under the assumption that* **SIVP** $\notin$ BQP *or* **GapSVP** $\notin$ BQP, *for* $2\sqrt{d} \le \gamma(d) \le d^{O(1)}$, $\beta(d) = d^{-O(1)}$, *and* $n(d) \in d^{O(1)}$, *and any family of PPT adversaries* $\{\mathcal{A}_d\}_{d\in\mathbb{N}}$, *there exists* $\mu(d) \in \mathsf{negl}(d)$ *such that*

$$\left| \Pr\left[\begin{matrix} \mathcal{A}_d(s_d) = 1 : \\ s_d \leftarrow\!\$\ \mathcal{N}(\mathbf{0}, I_d)^{\otimes n(d)} \end{matrix} \right] - \Pr\left[\begin{matrix} \mathcal{A}_d(s_d) = 1 : \\ \mathbf{w}_d \leftarrow\!\$\ \mathbb{S}^{d-1} \\ s_d \leftarrow\!\$\ \sqrt{2\pi}\mathcal{H}_{\mathbf{w}_d,\beta(d),\gamma(d)}^{\otimes n(d)} \end{matrix} \right] \right| \le \mu(d), \tag{7}$$

where $\otimes n(d)$ *represents the Cartesian product of* $n(d)$ *identical distributions. Equivalently, each observation* $\mathbf{r}_i$ $(1 \le i \le n(d))$ *is independently sampled from the identical distribution of either* $\mathcal{N}(\mathbf{0}, I_d)$ *or* $\sqrt{2\pi}\mathcal{H}_{\mathbf{w}_d,\beta(d),\gamma(d)}$ *[7].*

3 Problem Setup

With the hCLWE distribution (Definition 2.5), which is computationally indistinguishable from the multivariate Gaussian distribution, we construct the *Gaussian pancake mechanism (GPM)* as an analogy to GM. In particular, GPM perturbs the query result $q(D)$ by sampling the additive noise from the hCLWE distribution instead of the multivariate Gaussian distribution, as formalized in Definition 3.1.

Definition 3.1 (Gaussian pancake mechanism). *Given a query* $q : \mathcal{D} \to \mathbb{R}^d$ *and hyperparameters* $\sigma > 0$, $\mathbf{w} \in \mathbb{S}^{d-1}$, *and* $\beta, \gamma > 0$, *the* Gaussian pancake mechanism *is given by*

$$\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D) = q(D) + \sqrt{2\pi}\sigma \cdot \mathcal{H}_{\mathbf{w},\beta,\gamma}. \tag{8}$$

We denote the p.d.f. of $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D)$ *as* $f_{\sigma,\mathbf{w},\beta,\gamma}(\cdot; D)$.

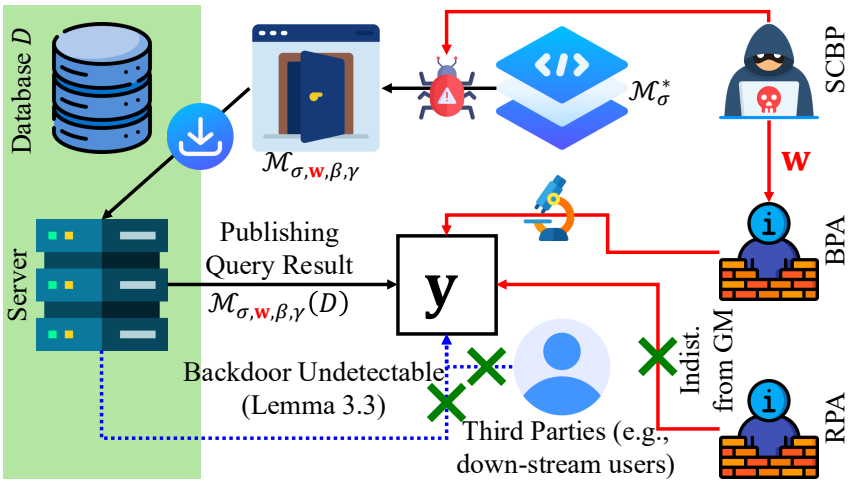


Fig. 2. Illustration of the Backdoor Attack Model: the supply-chain backdoor planter (SCBP) plants the backdoor by replacing GM with GPM. Consequently, the backdoor privacy attacker (BPA) gains information on D using GPM's output $\mathbf{y}$ and knowledge of the secret backdoor key $\mathbf{w}$. However, against a regular privacy attacker (RPA) without knowledge of $\mathbf{w}$, the GPM output is computationally indistinguishable from GM, thus preserving the same privacy guarantees. For the same reason, neither the server nor other parties (e.g., downstream users) can detect the existence of the backdoor.

Note that the multiplicative factor of $\sqrt{2\pi}$ in the hCLWE noise ensures that the scale remains consistent with GM. In the rest of this section, we formalize a backdoor attack based on GPM, while formalizing its undetectability by any party other than the backdoor key.

3.1 Threat Model

We consider the threat model presented in Figure 2, where a **supply-chain backdoor planter (SCBP)** installs the backdoor in the compromised software, alongside two types of PPT privacy attackers: the **backdoor privacy attacker (BPA)** and the **regular privacy attacker (RPA)**. Additionally, we assume that the **server and third parties (e.g., downstream users) monitor anomalies in the server's outputs**. Specifically, the SCBP, the BPA and the RPA follow the attack vector below:

- Given online software that utilizes differential privacy (e.g., Apple macOS, Amazon AWS, and Microsoft Azure), the SCBP targets the implementation of the authentic GM $\mathcal{M}^*_\sigma : \mathcal{D} \to \mathbb{R}^d$. It computes $\beta = \beta(d)$ and $\gamma = \gamma(d)$ that satisfy the conditions in Theorem 2.7, samples $\mathbf{w} \xleftarrow{\$} \mathbb{S}^{d-1}$ (acting as the secret **backdoor key**), plants the backdoor by replacing the authentic $\mathcal{M}^*_\sigma$ with GPM $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}$. Then, it shares the backdoor key $\mathbf{w}$, as well as the other configuration parameters β and γ, with the BPA.
- The server, in possession of a sensitive database D, downloads the compromised software. However, the compromised software is not open-source or the server fails to fully inspect the source code. It executes GPM as $\mathbf{y} \xleftarrow{\$} \mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D)$, and releases $\mathbf{y}$. Given the privacy requirements, we do **not** assume that the compromised software has access to the sensitive database D and the unperturbed query results $q(D)$.

Example 3.2. *If the server stores D and computes $q(D)$ in a protected environment while sampling the additive Gaussian noise from a randomness library, then compromising the randomness library by replacing the multivariate Gaussian distribution with hCLWE is sufficient to implant the backdoor,*

as the output will be equivalent to GPM. However, in this scenario, the randomness library need not (and should not) have access to D and $q(D)$.

- The BPA, with knowledge of the backdoor key $\mathbf{w}$, computes additional information about D from $\mathbf{y}$. Meanwhile, the RPA also attempts to compute additional information about D but without the knowledge of the backdoor key $\mathbf{w}$.

Note that the *SCBP could be the software developer itself* or other malicious attackers. Specifically, in addition to the prevalent numerical issues in DP libraries [9, 22, 28, 29, 39], the following real-world incidents showcase the prevalence and severity of the risk that DP and other privacy-sensitive software could be compromised, either via official channels or through software piracy and DNS hijacking:

- **Early versions of MacOS**, despite the deployment of DP mechanisms for user privacy protection, were found to have $\varepsilon \geq 16$, which offers only limited protection [56].
- The Lazarus Group committed multiple cryptocurrency thefts by utilizing **compromised software** loaded with **incorrect keys and malicious smart contracts** [51].
- A **backdoor** was suspected to have been inserted into **a published pseudorandom number generator standard**, allowing adversaries to readily decrypt materials encrypted using it [6].

Additionally, unlike the scenarios in Example 3.3, the compromised software has the same behaviour as the authentic one, except for the different yet indistinguishable noise distribution. Therefore, the server does not observe abnormalities in the execution trace that could lead to the detection of the backdoor.

Example 3.3. *Consider the following alternative methods of compromising the software and breaking the privacy guarantees:*

- *If the compromised software employs steganography [27] to embed additional information about D or $q(D)$ into the output $\mathbf{y}$, then additional reading authorization for these protected entities shall be requested. This causes an abnormal access pattern in the scenario of Example 3.2.*
- *If the BPA exploits the random seed to recover the additive noise, then either **1)** the random seed is embedded in the software, which produces consistent noise and is therefore detectable, or **2)** the BPA needs to monitor the internal state of the server (e.g., timestamps and/or all calls to the pseudorandom generator) in real time to compute the random seed, which is unrealistic in general and defendable by a trivial reset on the server's side.*

3.2 Undetectability of Backdoor

In this section, we characterize the covertness of the GPM under the threat model introduced in Section 3.1. In particular, Theorem 3.4 captures the case where the server downloads compromised software containing GPM $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}$ after the SCBP samples $\mathbf{w}$. The server then executes the mechanism **polynomially many** (rather than a single) times with input databases $D_1, D_2, \ldots, D_n$ (where $n = n(d)$ is a polynomial in d), respectively, and releases the results $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n$. We formalize that, by only examining these results, no party other than the SCBP and the BPA (i.e., one not knowing $\mathbf{w}$, including the server itself, downstream users, or any third party) can distinguish them from the authentic output of GM on the same sequence of input databases.

Theorem 3.4 (Covertness of GPM backdoor). *For any family of PPT adversaries $\{\mathcal{A}_d\}_{d\in\mathbb{N}}$, any $n(d) \in d^{O(1)}$, any $2\sqrt{d} \leq \gamma(d) \leq d^{O(1)}$, any $\beta(d) = d^{-O(1)}$, there exists $\mu(d) \in \mathsf{negl}(d)$ such that for any $D_1, D_2, \ldots, D_{n(d)} \in \mathcal{D}_d$,*

$$\left|\Pr\left[\begin{matrix}\mathcal{A}_d(o_d) = 1 : \\ o_d \leftarrow\!\!\$\ \mathsf{GMSeq}(d)\end{matrix}\right] - \Pr\left[\begin{matrix}\mathcal{A}_d(o_d) = 1 : \\ o_d \leftarrow\!\!\$\ \mathsf{GPMSeq}(d)\end{matrix}\right]\right| \leq \mu(d), \tag{9}$$

where $\mathsf{GMSeq}(d)$ and $\mathsf{GPMSeq}(d)$ are defined as Figure 3.

GMSeq(d)	GPMSeq(d)
1 : **for** $1 \le i \le n(d)$	1 : $\mathbf{w}_d \leftarrow\$ \mathbb{S}^{d-1}$
2 : $\quad \mathbf{y}_i \leftarrow\$ \mathcal{M}^*_{\sigma(d)}(D_i)$	2 : **for** $1 \le i \le n(d)$
3 : **endfor**	3 : $\quad \mathbf{y}_i \leftarrow\$ \mathcal{M}_{\sigma(d),\mathbf{w}_d,\beta(d),\gamma(d)}(D_i)$
4 : **return** $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n(d)}$	4 : **endfor**
	5 : **return** $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n(d)}$

Fig. 3. Definitions of GMSeq(d) and GPMSeq(d).

In Theorem 3.4, $\mathcal{A}_d$ represents any party (e.g., the server and downstream users) trying to distinguish between the outputs of GM and GPM, without knowledge of the backdoor key $\mathbf{w}$. Meanwhile, GMSeq(d) corresponds to the execution of the authentic GM, where the server applies GM multiple times on a sequence of input databases $D_1, D_2, \ldots, D_{n(d)}$, yielding outputs $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n(d)}$, respectively. Meanwhile, GPMSeq(d) corresponds to the planting and execution of GPM, where in Line 1, the SCBP samples its secret backdoor key $\mathbf{w}$ upon compromising the DP library. Subsequently, in Lines 2 to 4, the server executes GPM to output $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n(d)}$ on the same sequence of input databases. As long as GPM is executed polynomially many times (i.e., $n(d)$ is polynomially bounded), no party can efficiently distinguish its outputs from those of the authentic GM.

Proof sketch of Theorem 3.4. Assume the opposite. Then an efficient distinguishing adversary could distinguish between samples from the multivariate Gaussian distribution and the hCLWE distribution, thereby breaking the hardness assumption. □

Similar to Theorem 3.4, for the RPA without knowledge of the backdoor key $\mathbf{w}$, GPM maintains the same level of privacy guarantee, which is formalized in the appendix.

4 Analysis on the Statistical Privacy Leakage to the BPA

By Theorem 3.4, GPM is computationally indistinguishable from GM and achieves the same level of privacy computationally, i.e., against a PPT adversary without knowledge of the backdoor key $\mathbf{w}$. However, as shown in Figure 4, in the direction of $\mathbf{w}$, the probability density of GPM on neighbouring databases could be well separated, which creates an opportunity for the BPA who knows $\mathbf{w}$. Therefore, in this section, we proceed to show the actual statistical privacy loss, which is arbitrarily larger than that of GM. In particular, as shown in Figure 4, for neighbouring databases $D \simeq D'$, due to the differences between the centres $q(D)$ and $q(D')$, the peaks of the GPM probability densities may be misaligned, which therefore voids the DP guarantee provided by GM, despite the computational indistinguishability. Nevertheless, as GPM is still composed of multiple Gaussian components, it can be expected that, although significantly degraded compared with GM, some statistical DP guarantees can still be offered.

4.1 Lower Bound

To exactly analyze the additional privacy leakage arising from the misalignments of the peaks of probability densities, we first note that, by Lemma 2.6, the marginal distributions of the Gaussian components have the narrowest peaks when projected onto the secret direction $\mathbf{w}$. Therefore, as depicted in Figure 4, when executing the queries on different databases, the peaks of probability density functions could be significantly misaligned. This misalignment of the peaks causes additional privacy leakage compared to GM, where the privacy cost originates from the shift of the means in the Gaussian distributions [14, 16], and can thus be exploited by the BPA with knowledge of $\mathbf{w}$. To

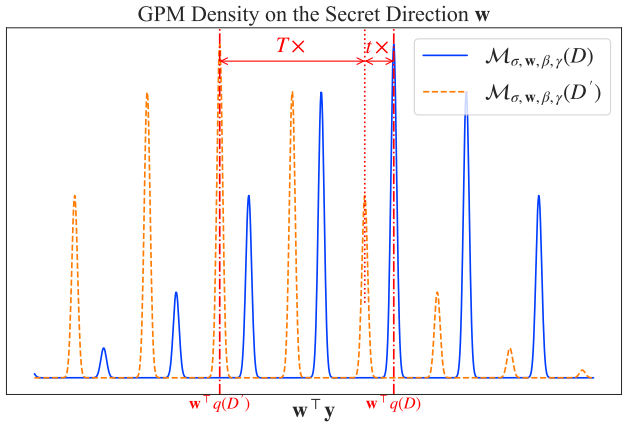


Fig. 4. When projected onto the secret direction $\mathbf{w}$, the probability densities of $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D)$ and $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D')$ for neighbouring databases $D \simeq D'$ are concentrated in disjoint sets of intervals and are therefore well separated, breaking the original DP guarantee. In this example, $q(D)$ and $q(D')$ differ by 2.4 peak widths, such that $T = 2$ and $t = 0.4$.

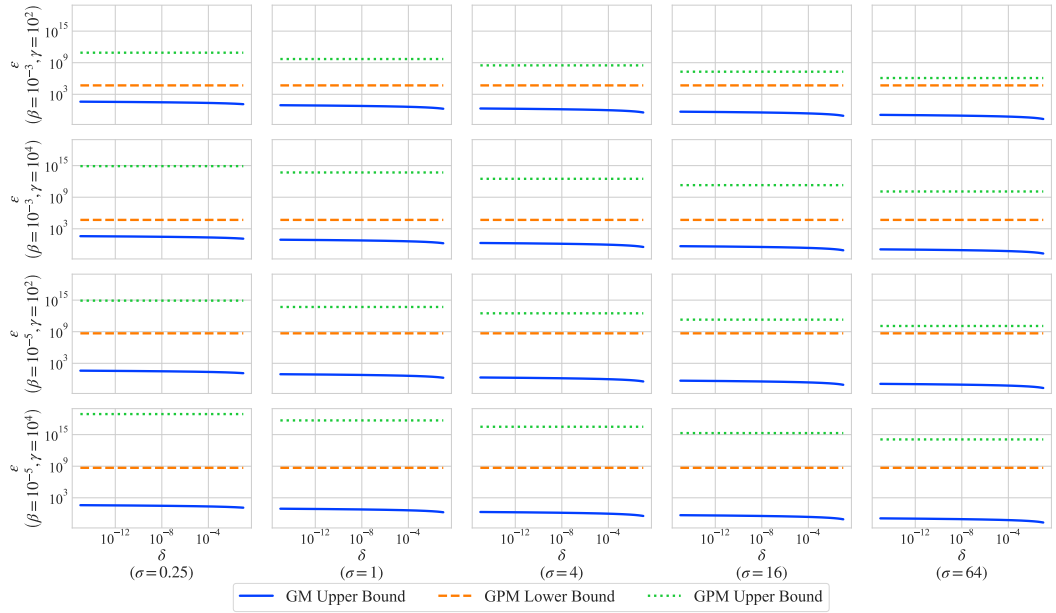


Fig. 5. Comparison between GM and GPM's privacy costs at $t = 0.25$, as in Theorem 4.2. Although computationally indistinguishable from GM, GPM's actual privacy costs exceed $\varepsilon = 10^4$ even when $\delta = 0.1$, which is infeasible for practical applications. Note that the lower and upper bounds of GPM's privacy costs still change with respect to δ, although at smaller relative scales.

simplify the analysis, we first project $\mathcal{H}_{\mathbf{w},\beta,\gamma}$ onto the direction of $\mathbf{w}$, which is equivalent to $\mathcal{H}_{1,\beta,\gamma}$,

with p.d.f.

$$\psi_{1,\beta,\gamma}(y) \propto \sum_{z\in\mathbb{Z}} a_{\beta,\gamma,z}\phi\left(y; \frac{\gamma z}{\beta^2+\gamma^2}, \frac{1}{2\pi}\cdot\frac{\beta^2}{\beta^2+\gamma^2}\right), \tag{10}$$

and state the key technical Lemma 4.1 for analyzing the behaviour of GPM in the direction of $\mathbf{w}$. As exemplified in Figure 4, T and t correspond to the integer and decimal parts of the ratio between the projected difference of the query result $\mathbf{w}^\top(q(D)-q(D'))$ and CLWE peak separation $\sqrt{2\pi}\sigma\cdot\frac{\gamma}{\beta^2+\gamma^2}$. Note that the decomposition into T and t is unique. The proof of Lemma 4.1 is provided in the appendix.

Lemma 4.1. *For any $T\in\mathbb{Z}$ and $t\in[-0.5,0.5)$, let $Y\sim\mathcal{H}_{1,\beta,\gamma}$ and $Y'\sim\mathcal{H}_{1,\beta,\gamma}+(T+t)\frac{\gamma}{\beta^2+\gamma^2}$ be two random variables. There exists a set $A(t):=\bigcup_{z\in\mathbb{Z}} I_z(t)$, where $I_z(t):=\left(\frac{\gamma\left(z-\frac{|t|}{2}\right)}{\beta^2+\gamma^2}, \frac{\gamma\left(z+\frac{|t|}{2}\right)}{\beta^2+\gamma^2}\right)$, such that*

$$\Pr\left[Y\in A(t)\right] \geq 1-2\Phi\left(-\frac{\gamma|t|}{\beta}\sqrt{\frac{\pi}{2(\beta^2+\gamma^2)}}\right), \tag{11}$$

$$\Pr\left[Y'\in A(t)\right] \leq 2\Phi\left(-\frac{\gamma|t|}{\beta}\sqrt{\frac{\pi}{2(\beta^2+\gamma^2)}}\right), \tag{12}$$

where Φ is the c.d.f. of the standard Gaussian distribution.

Lemma 4.1 can be applied to the marginal distributions of GPM's outputs on D and D' when projected onto $\mathbf{w}$, which gives rise to the lower bound of the actual statistical privacy leakage of GPM as stated in Theorem 4.2. The full proof of Theorem 4.2 is provided in the appendix.

Theorem 4.2. *For any $\sigma>0$, $\mathbf{w}\in\mathbb{S}^{d-1}$, $0<\beta<1$, $\gamma>1$, and any pair $T\in\mathbb{Z}$ and $t\in[-0.5,0.5)$, if there exists a pair of neighbouring databases $D\simeq D'$ such that $\sqrt{2\pi}\sigma\cdot(T+t)\frac{\gamma}{\beta^2+\gamma^2}=\mathbf{w}^\top(q(D')-q(D))$, then $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}$ is **not** (ε,δ)-DP for any*

$$0<\varepsilon<\underline{\varepsilon}:=\log\left(\frac{1-\delta}{2\Phi\left(-\frac{\gamma|t|}{\beta}\sqrt{\frac{\pi}{2(\beta^2+\gamma^2)}}\right)}-1\right) \tag{13}$$

and $0<\delta\leq 0.5$.

Proof sketch of Theorem 4.2. For neighbouring databases $D\simeq D'$, when $q(D)$ and $q(D')$ differ by $T+t$ in the direction of $\mathbf{w}$, the projected distributions of $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D)$ and $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D')$ are $\mathcal{H}_{1,\beta,\gamma}$ and $\mathcal{H}_{1,\beta,\gamma}+(T+t)\frac{\gamma}{\beta^2+\gamma^2}$, respectively. Therefore, by Lemma 2.6, to achieve (ε,δ)-DP, it must hold that $\Pr\left[Y\in A(t)\right]\leq e^\varepsilon\Pr\left[Y'\in A(t)\right]+\delta$, where Inequality (13) does not hold. □

Furthermore, note that the indistinguishability between GM and GPM hinges on the uniform randomness of the parameter $\mathbf{w}\leftarrow\!\$\,\mathbb{S}^{d-1}$. When the separation $\sqrt{2\pi}\sigma\cdot\frac{\gamma}{\beta^2+\gamma^2}$ between the Gaussian components is sufficiently small, the range of $T+t$ is significantly larger than 1. Therefore, the distribution of the decimal part t is close to the uniform distribution, such that in the average case, $|t|$ becomes a constant in Inequality (13). We formalize this in Proposition 4.3 and provide the detailed analysis in the appendix.

Proposition 4.3. *For any $\sigma>0$, $0<\beta<1$, and $\gamma>1$ such that $\gamma\gg\sqrt{d}\cdot\frac{\sigma}{\Delta}$, with constant probability over sampling $\mathbf{w}\leftarrow\!\$\,\mathbb{S}^{d-1}$, $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}$ is not (ε,δ)-DP where $\varepsilon\in\Theta\left(\frac{1}{\beta^2}\right)$ for any $0<\delta\leq 0.5$.*

However, it is also noteworthy to consider the marginal case where $q(D)$ and $q(D')$ are very close in the direction of $\mathbf{w}$, such that their separation is significantly smaller than one peak. This results in $T = 0$ **and** $t \ll 1$, which decreases the lower bound of privacy cost by Theorem 4.2, and is therefore a situation deemed **"unfavourable" from the perspective of the BPA**.

4.2 Upper Bound

Despite the broken original privacy guarantee against the BPA with knowledge of $\mathbf{w}$ shown in Section 4.1, GPM still achieves (statistical) DP theoretically, albeit at a significantly higher privacy cost. The intuition behind developing this upper bound is that, due to its nature as a mixture of Gaussian components with the same variance, GPM can still be viewed as first adding Gaussian noise and then post-processing by choosing the mean (i.e., deciding which component it falls into). However, the Gaussian noise is significantly narrower in the direction of $\mathbf{w}$ than in the authentic GM, thereby incurring a significantly larger privacy cost.

Theorem 4.4. *$\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(\cdot)$ is (ε,δ)-DP for any $0 < \delta \le 0.5$ such that $\varepsilon \in \Theta\left(\frac{\gamma^2\Delta^2}{\beta^2\sigma^2} + \frac{\gamma\Delta}{\beta\sigma}\sqrt{\log\frac{1}{\delta}}\right)$.*

Proof Sketch of Theorem 4.4. Note that $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D)$ is equivalent to

$$q(D) + \sqrt{2\pi}\sigma\left(\mathcal{N}\left(\mathbf{0}, \Sigma_{\mathbf{w},\beta,\gamma}\right) + \frac{\gamma\mathbf{w}\cdot\mathcal{N}_{\mathbb{Z}}\left(0, \frac{\beta^2+\gamma^2}{2\pi}\right)}{\beta^2+\gamma^2}\right), \tag{14}$$

where the discrete Gaussian noise can be viewed as post-processing. On the other hand, $\sqrt{2\pi}\sigma \cdot \mathcal{N}\left(\mathbf{0}, \Sigma_{\mathbf{w},\beta,\gamma}\right)$ has variance at least $\frac{\beta^2}{\beta^2+\gamma^2}\sigma^2$ in any direction (the lower bound is achieved in the direction of $\mathbf{w}$). Therefore, by Theorem 2.3, $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D)$ is (ε,δ)-DP for any

$$\varepsilon \ge \bar{\varepsilon} := \frac{\left(\beta^2+\gamma^2\right)\Delta^2}{2\beta^2\sigma^2} - \frac{\sqrt{\beta^2+\gamma^2}\Delta}{\beta\sigma}\Phi^{-1}(\delta) \tag{15}$$

and $0 < \delta \le 0.5$. □

We present the full proof of Theorem 4.4 in the appendix. As a sanity check, note that since Proposition 4.3 requires $\gamma \gg \sqrt{d}\cdot\frac{\sigma}{\Delta} \ge \frac{\sigma}{\Delta}$, it follows that $\frac{\gamma^2\Delta^2}{\beta^2\sigma^2} \gg \frac{1}{\beta^2}$. That is, the upper bound is higher than the lower bound. Moreover, we plot in Figure 5 the privacy costs under several configurations of hyperparameters at sensitivity $\Delta = 1$. In particular, it can be observed that the value of β is the determinant factor of the lower bound of GPM's privacy cost, while γ has little impact on it. On the other hand, the upper bound depends on both γ and β. These observations align with Proposition 4.3 and Theorem 4.4. For all choices of hyperparameters listed, the actual value of ε is at least 10^4, which indicates that no meaningful privacy protection can be achieved with GPM against an adversary that has knowledge of $\mathbf{w}$.

5 Distinguishing Attacks

Section 4 shows that the actual privacy cost of GPM is asymptotically larger than that of the genuine GM, despite their computational indistinguishability. In this section, we further demonstrate that this privacy leakage translates to additional power for the BPA with knowledge of the backdoor key $\mathbf{w}$. In particular, we construct a concrete attack to distinguish between neighbouring databases D_0 and D_1, from which the "privatized" query results are computed. The intuition behind this attack is that, as depicted in Figure 4, since the GPM densities are concentrated around several peaks in the

$\mathcal{A}_{\sigma,\mathbf{w},\beta,\gamma}(D_0, D_1, \mathbf{y})$

1 : **for** $i \leftarrow \{0,1\}$

2 : $\quad z_i \leftarrow \dfrac{\left(\beta^2+\gamma^2\right)\cdot\left(\mathbf{y}-q\left(D_i\right)\right)^\top \mathbf{w}}{\sqrt{2\pi}\sigma\cdot\gamma}$

3 : **endfor**

4 : **return** $i^* \leftarrow \underset{i\in\{0,1\}}{\arg\min}|z_i - \lfloor z_i \rceil|$

Fig. 6. Definition of $\mathcal{A}_{\sigma,\mathbf{w},\beta,\gamma}$.

direction of $\mathbf{w}$, the correct input database can be determined by checking whether the output is close to one of the peaks with very high probability.

Following the same setting for distinguishing attacks introduced in Section 2.3.1, we consider the probability that the BPA $\mathcal{A}_{\sigma,\mathbf{w},\beta,\gamma}$ (equipped with knowledge of $\mathbf{w}$) successfully identifies the index of the input database $i \in \{0,1\}$, i.e.,

$$\Pr\begin{bmatrix} i^* = i : \\ i \leftarrow\!\$\ \{0,1\} \\ \mathbf{y} \leftarrow\!\$\ \mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}(D_i) \\ i^* \leftarrow\!\$\ \mathcal{A}_{\sigma,\mathbf{w},\beta,\gamma}\left(D_0, D_1, \mathbf{y}\right) \end{bmatrix}, \tag{16}$$

where $i \in \{0,1\}$ is the real index of the database, and i^* is the guess of i by the attacker after observing the output of the mechanism.

By Lemma 2.6, the probability mass of $\mathcal{M}_{\sigma,\mathbf{w},\beta,\gamma}\left(D_i\right)$ is concentrated around the hyperplanes

$$H_i := \bigcup_{z\in\mathbb{Z}} \left\{ \mathbf{y} \in \mathbb{R}^d : \frac{\left(\mathbf{y}-q\left(D_i\right)\right)^\top \mathbf{w}}{\sqrt{2\pi}\sigma} = \frac{\gamma z}{\beta^2+\gamma^2} \right\}, \tag{17}$$

for $i \in \{0,1\}$, such that $\mathcal{A}_{\sigma,\mathbf{w},\beta,\gamma}\left(D_0, D_1, \mathbf{y}\right)$ (Figure 6) can be constructed by determining whether the closest hyperplane to $\mathbf{y}$ belongs to H_0 or H_1.

Theorem 5.1. *For any pair of neighbouring databases $D_0 \simeq D_1$, and the unique pair of $T \in \mathbb{Z}$ and $t \in [-0.5, 0.5)$ such that $\sqrt{2\pi}\sigma\cdot(T+t)\frac{\gamma}{\beta^2+\gamma^2} = \mathbf{w}^\top\left(q(D_1)-q(D_0)\right)$, the probability that $\mathcal{A}_{\sigma,\mathbf{w},\beta,\gamma}$ correctly identifies the input database (i.e., Equation (16)) is at least $1-2\Phi\left(-\frac{\gamma|t|}{\beta}\sqrt{\frac{\pi}{2(\beta^2+\gamma^2)}}\right)$.*

Proof of Theorem 5.1. Without loss of generality, condition on $i=0$. Observe that, when $\mathbf{y}$ is closer to H_0 than H_1, $\mathcal{A}_{\sigma,\mathbf{w},\beta,\gamma}$ outputs 0. Following the same argument as in Theorem 4.2, this event occurs with probability at least $1-2\Phi\left(-\frac{\gamma|t|}{\beta}\sqrt{\frac{\pi}{2(\beta^2+\gamma^2)}}\right)$. □

By Theorem 5.1, the success rate of the distinguishing attack is very close to 1 given $\gamma \gg 1 \gg \beta$, which we further verify empirically in Section 7.1. Here, the failure probability corresponds to the case where the output shifts away from the peaks of the GPM density, and instead falls closer to the peaks corresponding to the other database, which happens with infinitesimal probability under proper choices of parameters.

6 Mitigations

In this section, we discuss possible mitigations against the privacy attack enabled by GPM as one major possible direction for future work. In particular, in Section 6.1, we discuss a possible mitigation

method under the same threat model as considered in Section 3.1, which is fully compatible with the scenario discussed in Example 3.2, where the randomness library is compromised. Meanwhile, in Section 6.2, we discuss viable methods of enforcing the correct implementations and executions of the DP mechanisms. We present in the appendix an additional mitigation method under the setting of distributed DP with multiple servers, where a subset of the servers executes the compromised software with GPM.

6.1 Random Rotation

Given that the privacy attack hinges on the BPA's knowledge of $\mathbf{w}$, and that the output is computationally indistinguishable from GM otherwise, one mitigation against GPM is to apply a random rotation to the sampled noise. Specifically, as described in Algorithm 1, upon sampling the noise value $\mathbf{r}$ from the possibly backdoored multivariate Gaussian distribution $\mathcal{R}$, the server does not directly add the noise to the unperturbed query result $q(D)$. Instead, it rotates $\mathbf{r}$ to a direction $\mathbf{u}$ sampled uniformly at random, only preserving its length $\|\mathbf{r}\|$. Equivalently, the noise added to the query result is $\sigma \cdot \|\mathbf{r}\| \cdot \mathbf{u}$.

Algorithm 1 Noise-Rotated Gaussian Mechanism (NRGM)

Require: Database D; query q; variance σ; d-dimensional Gaussian distribution $\mathcal{R}$ (possibly replaced with hCLWE).

1: $\mathbf{r} \leftarrow\!\!\$\ \mathcal{R}$ ▷ Sample the noise
2: $\mathbf{u} \leftarrow\!\!\$\ \mathbb{S}^{d-1}$ ▷ Sample a random direction
3: **return** $\mathbf{y} \leftarrow q(D) + \sigma \cdot \|\mathbf{r}\| \cdot \mathbf{u}$.

Algorithm 1 does not affect the output distribution if the mechanism has not been backdoored, and can be executed in $O(d)$ time and space complexity. However, if the mechanism is indeed backdoored as GPM, the equivalent distribution of the output $\mathbf{y}$ is $\mathcal{M}_{\sigma,\mathbf{w}',\beta,\gamma}$ for another uniformly random $\mathbf{w}'$ that is not known to the BPA. Therefore, the noise-rotated GPM maintains the same level of privacy guarantee as GM, as formalized in the appendix. Nevertheless, recall that the backdoor planted in the randomness of GPM can also be transformed into a backdoor in the randomness of sampling the random direction $\mathbf{u}$. Therefore, $\mathbf{r}$ and $\mathbf{u}$ should be sampled using different libraries. To prevent this cat-and-mouse game, we also consider adding further verification to fundamentally mitigate the existence of the backdoor.

6.2 Verifiable DP

The integrity of GM may also be enforced by adding further verification to identify deviations from the authentic GM and unintended privacy leakages.

6.2.1 Cryptographic Proofs. The execution of GM can be verified by zero-knowledge proofs (ZKPs), cryptographic primitives that certify the correct execution of prescribed computations [57]. With ZKPs applied, an external verifier can be convinced that the authentic GM, rather than GPM, is genuinely executed by the server, such that no backdoor is planted in the server's output [5, 31, 32, 40, 53, 55]. However, the application of ZKPs incurs fundamental overhead, which poses significant challenges to their practical implementation.

6.2.2 Formal Verifications. Formal verification techniques can identify bugs in the DP mechanisms implemented as computer programs, with the purpose of detecting unintended privacy leakages [41, 50, 59]. Such techniques can automatically identify the GPM backdoor as a breach of the intended privacy guarantee. However, this requires the open-sourcing of the DP libraries and the faithful execution of the verified programs.

7 Experiments

In this section, we conduct experimental evaluations on GPM, with a focus on examining the power of distinguishing attacks gained by the BPA with knowledge of the backdoor key $\mathbf{w}$, as described in Section 5. All experiments were conducted on a computing node equipped with an NVIDIA A40 with 48GB VRAM, which shares 2 AMD EPYC 7272 12-Core CPUs and 512 GB RAM with 7 other computing nodes of the same hardware configuration. Unless specified otherwise, all experiments were repeated 100 times. The open-source code and raw experiment logs are available at https://github.com/jvhs0706/GPM.

7.1 Experiments on Distinguishing Attacks

Since the covertness guarantees in Section 3.2 hinge on high dimensionality, it is only meaningful to consider additional privacy leakage under sufficiently large d (e.g., $d \geq 256$) such that the original DP guarantees continue to hold against the RPA. Therefore, in this section, we report the success rates of distinguishing attacks; that is, the percentage of distinguishing attacks in which the index of the selected database is correctly identified by the BPA ($i = i^*$), which approximates the frequency in Equation (16). We conducted experiments for distinguishing attacks on two classes of queries that are ubiquitously studied in the DP literature where GM applies and that have a large output dimension: namely, DP histogram (DP-hist) queries [3, 23, 34, 60] and the gradient computation in DP stochastic gradient descent (DP-SGD) [1, 17, 24].

- For DP-hist queries, we experiment with artificial datasets $D \in \{1, 2, \ldots, d\}^n$. That is, each of the n records is assigned one of the d classes. We set $d \in \{256, 4096, 65536\}$, such that the non-private query returns a d-dimensional vector $\mathbf{h}$ with the i-th element being $\mathbf{h}_i := \left|\left\{D_j : D_j = i\right\}\right|$, i.e., the count of the i-th class in the database. Two databases D and D' are defined as neighbouring if D is equivalent to D' after adding or removing one record. Therefore, the histogram query has a sensitivity of 1 under the ℓ^2 norm.
- For the gradient computation in DP-SGD, we follow the algorithm of the original paper [1] and the same setting as the distinguishing attacks in previous studies on DP numerical issues [29]. In particular, we experiment with the MNIST dataset using the LeNet-5 structure, and the CIFAR-10 dataset using VGG19, ResNet50, and MobileNet V2 structures, which have $2 \times 10^6 \leq d \leq 4 \times 10^7$ parameters, respectively. Given a batch of $n = 128$ data points $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, we compute the partial gradient of the loss function with respect to each data point $\mathbf{g}_i$, clip each $\mathbf{g}_i$ to an ℓ^2 norm of $C \in \{4, 8\}$ (denoting the clipped gradient as $\mathbf{g}'_i$), and define the non-private query result as $\mathbf{g} := \frac{1}{n}\sum_{i=1}^n \mathbf{g}'_i$. The neighbouring relation is defined by replacing one data point in the batch. For fairness, we fix the random seeds for the training process to keep the non-replaced $\mathbf{g}'_i$s consistent for queries on the neighbouring databases. Therefore, the sensitivity of the query is $\frac{2C}{n}$ under the ℓ^2 norm. We also consider two phases during training: before the model has converged (training from a randomly initialized set of parameters), and after the model has converged (loading a set of pretrained parameters with sufficiently high accuracy).

For both classes of queries, we fix $\delta^* = 10^{-10}$ and vary ε^*, where $(\varepsilon^*, \delta^*)$ are the privacy parameters of the authentic GM as defined in Theorem 2.3, such that the scale of the noise σ can be computed accordingly. Note that, by Theorems 2.7 and 3.4, the computational indistinguishability result holds for any $\beta(d) \in d^{-O(1)}$ and $2\sqrt{d} \leq \gamma \leq d^{O(1)}$. Meanwhile, by Theorems 4.2 and 5.1, β is the decisive factor for the actual lower bound on privacy leakage and the success rates of the distinguishing attacks. Therefore, $\frac{1}{\beta}$ can be as large as **any polynomial**. Consequently, we choose β in our experiments to range between 10^{-5} and 10^{-1} while varying $256 \leq d \leq 4 \times 10^7$, ensuring that $\frac{1}{\beta}$ is at most on the order of d^2, which satisfies the conditions. Similarly, we constrain γ to

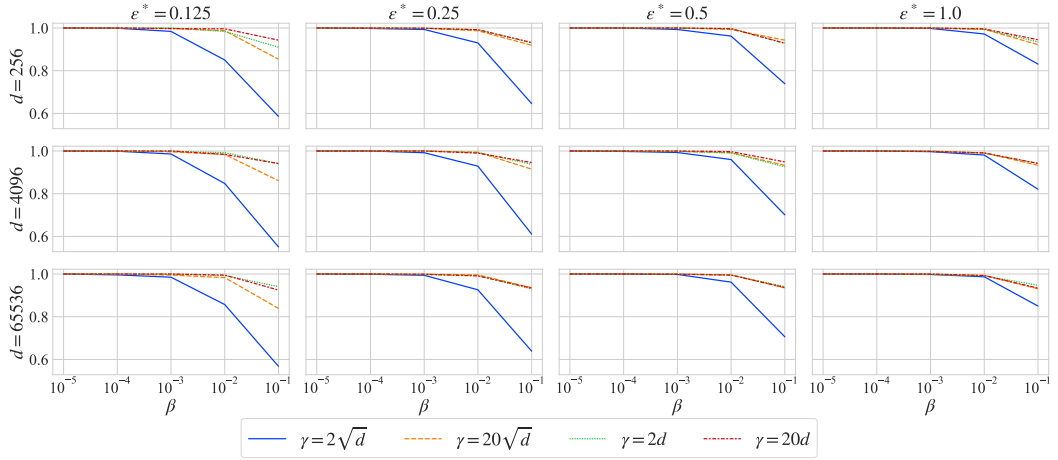


Fig. 7. Distinguishing attack success rates on DP-hist. Success rates approach 1 when $\beta < 10^{-3}$, and degrade significantly for $\beta \in \{10^{-1}, 10^{-2}\}$, especially for $\gamma = 2\sqrt{d}$. For larger β, increasing ε^* slightly improves performance.

a reasonable range between $2\sqrt{d}$ (the default value unless otherwise specified) and $20d$ in all experiments reported in this section.

It is also important to note that the purpose of the experiments on distinguishing attacks presented in this section is to showcase the seriousness of the GPM backdoor, and the evaluated attacks are independent of GM's numerical issues despite the similar experimental settings. In particular, the design and analysis of the distinguishing attack, as presented in Section 5, are fully principled in the real domain and therefore do not exploit vulnerabilities in floating-point arithmetic to increase the success rates.

7.1.1 DP-hist. As shown in Figure 7, for DP-hist queries, we observe that when $\beta \leq 10^{-3}$, distinguishing attacks succeed with a probability close to 1. On the other hand, for $\beta = 0.1$, the success rates are not significantly higher than those of random guessing, especially for the smallest value of $\gamma = 2\sqrt{d}$. This phenomenon is in accordance with Theorem 5.1 and is further visually explained in Figure 4, showing that the peaks, which are of width approximately $\sqrt{2\pi}\frac{\gamma}{\beta}$, are wider due to the larger values of β, resulting in greater overlaps in the probability masses. Therefore, the SCBP is motivated to select smaller values of β, as long as $\beta(d)$ is polynomially bounded as required by Theorem 3.4.

Furthermore, it can also be observed that, for larger values of β, larger ε^* values generally result in better success rates. This phenomenon is in accordance with the $(\varepsilon^*, \delta^*)$-DP in authentic GM, and can also be explained by the fact that larger ε^* values result in smaller σ values. Consequently, for neighbouring databases D and D' and the unique pair of $T \in \mathbb{Z}$ and $t \in [-0.5, 0.5)$ such that $\sqrt{2\pi}\sigma \cdot (T + t)\frac{\gamma}{\beta^2+\gamma^2} = \mathbf{w}^\top (q(D) - q(D'))$, the probability that $T = 0$ and $t \ll 1$ becomes lower over the randomness of sampling $\mathbf{w}$. By Theorem 5.1, this results in a higher success rate for distinguishing attacks. Similarly, note that for fixed σ, D, D' and $\mathbf{w}$, the product of $(T + t)$ and $\frac{\gamma}{\gamma^2+\beta^2}$ is constant. Consequently, the value of $(T + t)$ is approximately proportional to γ. Thus, for larger values of γ, $(T + t)$ generally takes on larger values; hence, the event where $T = 0$ and $t \ll 1$ becomes less likely. This explains why larger values of γ compensate for the drop in success rates for larger values of β.

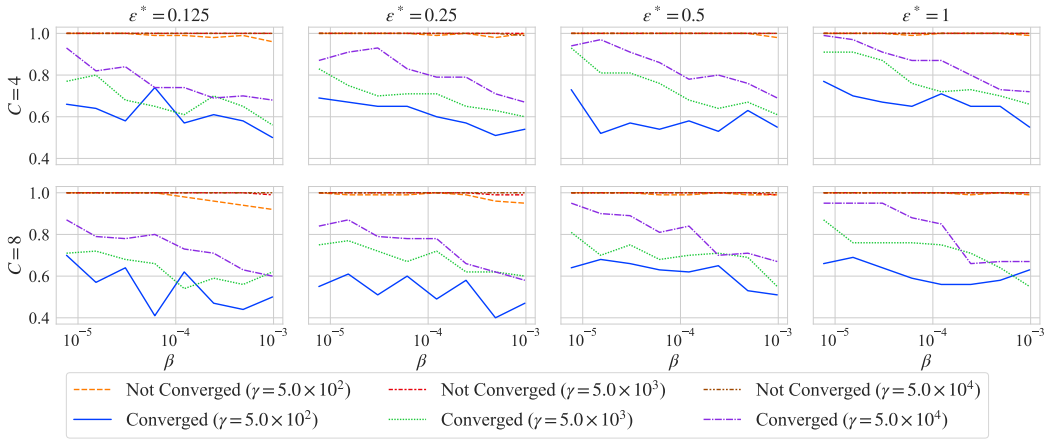


Fig. 8. Distinguishing attack success rates on DP-SGD with MNIST. Success rates drop significantly after model convergence.

7.1.2 DP-SGD. We report the experimental results of distinguishing attacks on DP-SGD with the MNIST dataset and LeNet-5 in Figure 8. A trend similar to that in Figure 7 can be observed if the model is trained from random initialization. That is, the success rate of the distinguishing attacks approaches 1 in all experiments when β is at the level of 10^{-5} to 10^{-4}, while at $\beta \approx 10^{-3}$, a slight drop in the success rate of 5% to 10% can be observed, and larger ε^* values, and therefore smaller σ values, tend to compensate for this drop. However, one unexpected phenomenon can be observed on the SGD steps after the model has converged: the success rates are consistently below 1 even if smaller values of β are chosen. This can be explained by the fact that, as the model has converged, the computed gradients afterwards are significantly smaller in magnitude [58]. That is, $\|q(D) - q(D')\| \ll \Delta$, such that in Theorem 5.1, $T = 0$ and $t \ll 1$ with sufficiently high probability, which impacts the overall success rate of the distinguishing attacks. Meanwhile, similar to the DP-hist case, increasing the value of γ counteracts the increase in this probability, thereby restoring the success rates after convergence to more than 90% for larger ε^* values and smaller β values.

We further report the experimental results on CIFAR-10 in Table 2. It can be observed that 100% success rates are achieved for all instances when $\beta = 10^{-5}$, and for most instances when $\beta = 10^{-4}$. A slight drop in success rate is observed when $\beta = 10^{-3}$, although it still remains at or above 97% in most cases. Moreover, no drop in success rate is observed after convergence. This phenomenon can be explained by the fact that the gradients and their differences remain of considerable magnitude, as illustrated in Table 3. This is attributed to the relatively lower accuracies on CIFAR-10, which result in loss functions that are not very close to 0, as well as the larger model sizes, which increase the magnitudes of the gradients. Consequently, the edge case in Theorem 5.1, where $T = 0$ and $t \ll 1$, no longer occurs with significant probability.

7.2 Additional Experiments

By Theorem 3.4, the GPM backdoor cannot be discovered by any PPT adversary without knowledge of the backdoor key, except with negligible probability. That is, it is impossible to distinguish the outputs of GM and GPM using any metric independent of $\mathbf{w}$, provided the metric can be computed efficiently. To further verify this property, we measure the actual ℓ^2 error, averaging over all instances of the experiments considered in Figure 7, and present the results in Table 4. It can be observed

Table 2. Distinguishing attack success rates on DP-SGD with CIFAR-10. Most instances achieve 100% success rates. Unlike Figure 8, no drop in success rate is observed after convergence compared with before convergence. Note that, as detailed in the appendix, repeating the experiments with larger values of $\gamma = 20\sqrt{d}$ and $\gamma = 200\sqrt{d}$ (in contrast to the default value of $\gamma = 2\sqrt{d}$ in this table) results in 99–100% success rates in all experimental instances.

Model	d	Converged	ε^*	$\gamma(\times 10^3)$	$\beta\,(C=4)$			$\beta\,(C=8)$		
					10^{-5}	10^{-4}	10^{-3}	10^{-5}	10^{-4}	10^{-3}
VGG19	39M	✗	0.125	12	100	100	99	100	99	96
			0.25		100	100	97	100	100	97
			0.5		100	100	100	100	100	100
			1		100	100	100	100	100	99
		✔ (94.0%)	0.125		100	100	97	100	99	97
			0.25		100	100	98	100	100	97
			0.5		100	100	100	100	100	97
			1		100	100	99	100	100	98
ResNet50	23M	✗	0.125	9.7	100	100	97	100	100	97
			0.25		100	100	97	100	100	97
			0.5		100	100	100	100	100	100
			1		100	100	100	100	100	100
		✔ (93.7%)	0.125		100	100	99	100	100	98
			0.25		100	100	98	100	100	99
			0.5		100	100	100	100	100	99
			1		100	100	100	100	100	100
MobileNetV2	2.2M	✗	0.125	3.0	100	98	97	100	100	98
			0.25		100	100	98	100	100	100
			0.5		100	100	100	100	99	100
			1		100	100	97	100	100	100
		✔ (93.9%)	0.125		100	100	98	100	99	93
			0.25		100	100	100	100	100	100
			0.5		100	100	99	100	100	98
			1		100	100	100	100	100	100

Table 3. Median magnitude of gradients and the difference between gradients computed on neighbouring databases, both of which are significantly larger in CIFAR-10 tasks regardless of the convergence status.

Task	Model	$\|q(D)\|$		$\|q(D) - q(D')\|$	
	Converged	✗	✔	✗	✔
MNIST	LeNet5	0.32	0.025	0.32	3.3×10^{-6}
CIFAR-10	VGG19	12	1.4	13	0.81
CIFAR-10	ResNet50	980	2.9	110	0.31
CIFAR-10	MobileNetV2	11	6.2	1.35	0.59

that the actual ℓ^2 error and the theoretically computed ℓ^2 error are not significantly different. This closeness in error magnitude serves as empirical evidence of the backdoor's undiscoverability.

Table 4. Comparison between GM's and GPM's actual ℓ^2 errors and the theoretical expectation computed from GM. No significant differences are observed.

	ε^*	0.125	0.25	0.5	1
$d = 256$	Expect	815.5	408.4	204.8	103.2
	GM	811.8	404.9	203.9	102.6
	GPM	819.1	407.8	206.1	102.2
$d = 4096$	Expect	3262.0	1633.5	819.3	412.1
	GM	3263.3	1636.1	819.9	411.4
	GPM	3260.3	1632.9	819.3	412.5
$d = 65536$	Expect	13048.1	6534.1	3277.0	1648.4
	GM	13043.7	6536.5	3277.2	1648.7
	GPM	13044.2	6526.8	3276.3	1648.3

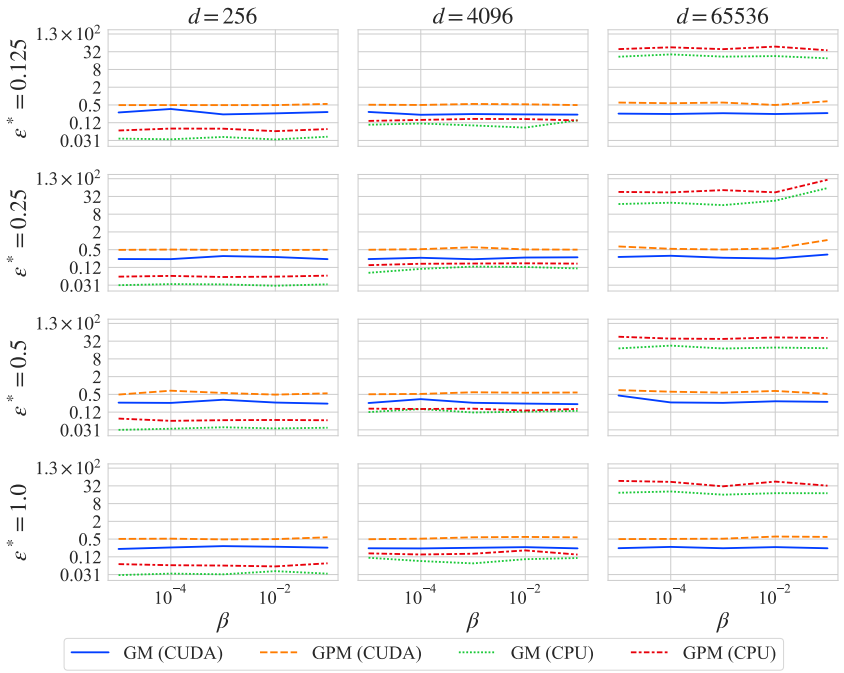


Fig. 9. Comparison of noise sampling times between GM and GPM (in milliseconds), measured on CPU and CUDA devices; GPM incurs a 0.5–1.5x slowdown. Note that GPM's implementation is not fully optimized. The running times of the other components (e.g., the unperturbed query processing time) are identical between GM and GPM, yet not included.

While we acknowledge the challenge of exactly matching the execution times of GM and GPM, we argue that the differences in running times are unlikely to expose the backdoor. In particular, Figure 9 compares the noise sampling times of GM and GPM under various DP-hist configurations; the results show that Gaussian pancake noise is sampled only 0.5–1.5x slower than standard Gaussian noise. We further note that this difference is a **pessimistic estimate** of GPM's covertness regarding this side channel. Specifically:

- Under the threat model described in Section 3.1, where the closed-source benign implementation is replaced by a malicious backdoored version in the software supply chain, the server lacks access to the benign software as a reference. Similarly, the server cannot obtain the exact implementation details of either the benign or the compromised software to derive an accurate estimate of execution time. Therefore, the 0.5–1.5x slowdown, which remains within the same order of magnitude, could be plausibly attributed to implementation variances or uncontrollable factors such as hardware load and cooling.
- The reported running time corresponds **only** to noise sampling, which is the sole source of the execution-time difference between GM and GPM. All running times other than noise sampling (e.g., the unperturbed query processing time) are excluded from Figure 9 and are identical for GM and GPM. In particular, differences in execution times would be diluted by the unperturbed query execution and any applicable post-processing. Consequently, we anticipate that the differences in end-to-end execution times between the two mechanisms will be less significant.

Also, unlike the implementation of GM, which enjoys low-level hardware accelerations, our implementation of GPM is not fully optimized at an industrial level. Specifically, GM was implemented using industrial-level APIs such as `torch.randn`, which have undergone hardware-level optimizations (e.g., compression into a single CUDA kernel and instruction set optimizations), whereas GPM relies on higher-level tensor operations (typically involving multiple CUDA kernel calls). Therefore, it is foreseeable that dedicated SCBPs could optimize GPM to achieve an even greater level of covertness with a smaller difference in running times.

8 Related Work

Numerical issues have historically been among the most common and exploitable software defects undermining the rigorous theoretical guarantees provided by differential privacy. In particular, though DP mechanisms are mathematically sound when formulated in the real-number domain, their floating-point implementations can subtly yet damagingly deviate from their original design, typically due to imprecisions in the least significant digits. Mironov [39] was the first to identify that floating-point imprecision in the Laplace mechanism led to unintended leakage, as the actual distribution of the least significant bits highly depends on the unperturbed query result and therefore leaks significantly more information about it than intended. As a result, privacy attacks were found to be successful against all four publicly available libraries evaluated. As a countermeasure, it was proposed to post-process the least significant bits to restore the intended DP guarantees. Similarly, Ilvento [28] demonstrated related risks in the Exponential mechanism, where rounding errors can result in some outputs being assigned zero probability, effectively setting the privacy cost to $\varepsilon = +\infty$. To address this, a base-2 variant was proposed to better align with hardware-level arithmetic in binary computers. More generally, Casacuberta et al. [9] showed that such vulnerabilities are more widespread than previously believed, particularly in basic operations like summation, where arithmetic discrepancies can cause the true sensitivity to be underestimated and insufficient noise to be added. These results underscore the importance of secure and precise implementations, even for theoretically well-understood DP mechanisms.

Most relevant to our work, Jin et al. [29] found similar vulnerabilities in the Gaussian mechanism across multiple standard sampling algorithms used in the implementations of the Gaussian mechanism [33, 37, 38], in addition to previously studied timing attacks [2, 22, 46]. In particular, similar to the Laplace mechanism, the least significant bits are also a source of unintended privacy leakage. Leveraging this vulnerability, they proposed distinguishing attacks that successfully exploited this software defect, which caused only a minor deviation from the theoretical design under which the originally sound DP guarantee holds. In response, patches have been introduced in IBM's

differential privacy library [25, 26] to obfuscate the least significant bits, which were shown to be especially vulnerable.

Goldwasser et al. [19] presented the first application of the CLWE problem [7] for constructing undetectable backdoors in deep learning models. Their work introduced several cryptographically motivated backdoor mechanisms, including those based on digital signatures [20], sparse PCA [4], and CLWE. In the CLWE-based construction, the authors considered Gaussian features extracted from a randomly initialized fully connected layer that approximates the Gaussian kernel [45]. These features are then used to train a binary linear classifier. The backdoor is planted by replacing the Gaussian distribution with a CLWE distribution during initialization, such that modifying the input along a secret vector $\mathbf{w}$ flips the classifier's prediction. This construction highlights the potential for CLWE to be applied beyond learning theory [10–13, 52].

Formal methods for verifying differential privacy have evolved significantly from foundational work on type systems for sensitivity analysis [47]. These systems introduced a functional language in which types track function sensitivity, enabling automatic proofs of differential privacy. Building on this foundation, Duet [41] proposed an expressive higher-order language and linear type system for verifying the privacy of general-purpose higher-order programs. CheckDP [59] advanced automated verification by introducing a unified framework that can both prove and disprove differential privacy claims. It is capable of finding subtle bugs and generating counterexamples that elude prior techniques. More recently, Reshef et al. [50] introduced local differential classification privacy (LDCP), a new privacy notion designed for black-box classifiers, extending the idea of local robustness into the DP domain. These efforts have inspired ongoing work in the formal auditing of modern DP systems [35, 44] in practical deployments.

Beyond formal verification of DP programs, a growing body of work has focused on making the execution of DP mechanisms themselves verifiable. This includes proof-carrying implementations of DP mechanisms in the Fuzz programming language [40], as well as verifiable instantiations of randomized response [31] and the floating-point Gaussian mechanism used in DP-SGD [53]. With the increasing reliance on distributed data analysis and secure multiparty computation, verifiability has also been explored in collaborative settings. Notable examples include verifiable DP protocols for Prio [32], the Binomial mechanism [5], and the discrete Laplace mechanism [55].

9 Conclusion

This study introduces the Gaussian pancake mechanism (GPM), a novel and practical construction of a backdoor attack against the Gaussian mechanism (GM), one of the most prevalent DP mechanisms in industrial applications. We have shown that, for a backdoor attacker with knowledge of the backdoor key, the privacy guarantee degrades arbitrarily from both theoretical and empirical perspectives, while the attack remains undetectable by the server and third parties. In light of these findings, we advocate for the use of open-source DP libraries, formal verification techniques, and cryptographic safeguards to mitigate such risks. Future research may further explore stronger yet more elusive adversarial manipulations of DP mechanisms that **1)** could result in exacerbated privacy loss (e.g., full recovery of the unperturbed query result $q(D)$, the input database D, or an end-to-end privacy attack against DP-SGD), and **2)** are strictly computationally undetectable, not only via checking the mechanism's outputs but also through side channels such as execution times. The purpose of developing such attacks should not be to weaken privacy, but to advance the robustness, verifiability, and trustworthiness of privacy-preserving technologies. To this end, we also encourage future work on more robust countermeasures that extend the noise rotation technique to reduce reliance on trusted external randomness, as well as the design of more efficient verification protocols to certify the absence of backdoors in practical, large-scale deployments.

Code and Extended Version Availability

To support reproducibility, the complete source code, raw experimental logs, and step-by-step reproduction instructions are publicly available at https://github.com/jvhs0706/GPM. The full, extended version of this paper, which includes the complete appendix, is available at https://arxiv.org/abs/2509.23834.

Acknowledgments

This work is supported by NSERC (Discovery and Alliance grants) and the Canada CIFAR AI Chairs program. We would like to thank Shubhankar Mohapatra, Shufan Zhang, and Karl Knopf for their insightful discussions, as well as the anonymous reviewers for their detailed and constructive comments, which helped improve the paper. Icons in Figure 2 were created by Freepik from Flaticon.

References

[1] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi (Eds.). ACM, 308–318. doi:10.1145/2976749.2978318

[2] Marc Andrysco, David Kohlbrenner, Keaton Mowery, Ranjit Jhala, Sorin Lerner, and Hovav Shacham. 2015. On Subnormal Floating Point and Abnormal Timing. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*. IEEE Computer Society, 623–639. doi:10.1109/SP.2015.44

[3] Raef Bassily and Adam D. Smith. 2015. Local, Private, Efficient Protocols for Succinct Histograms. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, Rocco A. Servedio and Ronitt Rubinfeld (Eds.). ACM, 127–135. doi:10.1145/2746539.2746632

[4] Quentin Berthet and Philippe Rigollet. 2013. Complexity Theoretic Lower Bounds for Sparse Principal Component Detection. In *COLT 2013 - The 26th Annual Conference on Learning Theory, June 12-14, 2013, Princeton University, NJ, USA (JMLR Workshop and Conference Proceedings, Vol. 30)*, Shai Shalev-Shwartz and Ingo Steinwart (Eds.). JMLR.org, 1046–1066. http://proceedings.mlr.press/v30/Berthet13.html

[5] Ari Biswas and Graham Cormode. 2023. Interactive Proofs For Differentially Private Counting. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023*, Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda (Eds.). ACM, 1919–1933. doi:10.1145/3576915.3616681

[6] Daniel R. L. Brown and Kristian Gjøsteen. 2007. A Security Analysis of the NIST SP 800-90 Elliptic Curve Random Number Generator. In *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings (Lecture Notes in Computer Science, Vol. 4622)*, Alfred Menezes (Ed.). Springer, 466–481. doi:10.1007/978-3-540-74143-5_26

[7] Joan Bruna, Oded Regev, Min Jae Song, and Yi Tang. 2021. Continuous LWE. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, Samir Khuller and Virginia Vassilevska Williams (Eds.). ACM, 694–707. doi:10.1145/3406325.3451000

[8] Clément L. Canonne, Gautam Kamath, and Thomas Steinke. 2020. The Discrete Gaussian for Differential Privacy. *CoRR* abs/2004.00010 (2020). arXiv:2004.00010 https://arxiv.org/abs/2004.00010

[9] Sílvia Casacuberta, Michael Shoemate, Salil P. Vadhan, and Connor Wagaman. 2022. Widespread Underestimation of Sensitivity in Differentially Private Libraries and How to Fix It. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi (Eds.). ACM, 471–484. doi:10.1145/3548606.3560708

[10] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru Zhang. 2023. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. https://openreview.net/forum?id=zyLVMgsZ0U_

[11] Ilias Diakonikolas, Daniel Kane, and Lisheng Ren. 2023. Near-Optimal Cryptographic Hardness of Agnostically Learning Halfspaces and ReLU Regression under Gaussian Marginals. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA (Proceedings of Machine Learning Research, Vol. 202)*, Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (Eds.). PMLR, 7922–7938. https://proceedings.mlr.press/v202/diakonikolas23b.html

[12] Ilias Diakonikolas, Daniel Kane, Lisheng Ren, and Yuxin Sun. 2023. SQ Lower Bounds for Non-Gaussian Component Analysis with Weaker Assumptions. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). http://papers.nips.cc/paper_files/paper/2023/hash/0d00a699f60e642b310eb04b76cc7731-Abstract-Conference.html

[13] Ilias Diakonikolas, Daniel M. Kane, Thanasis Pittas, and Nikos Zarifis. 2023. SQ Lower Bounds for Learning Mixtures of Separated and Bounded Covariance Gaussians. In *The Thirty Sixth Annual Conference on Learning Theory, COLT 2023, 12-15 July 2023, Bangalore, India (Proceedings of Machine Learning Research, Vol. 195)*, Gergely Neu and Lorenzo Rosasco (Eds.). PMLR, 2319–2349. https://proceedings.mlr.press/v195/diakonikolas23b.html

[14] Jinshuo Dong, Aaron Roth, and Weijie J. Su. 2019. Gaussian Differential Privacy. *CoRR* abs/1905.02383 (2019). arXiv:1905.02383 http://arxiv.org/abs/1905.02383

[15] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings (Lecture Notes in Computer Science, Vol. 3876)*, Shai Halevi and Tal Rabin (Eds.). Springer, 265–284. doi:10.1007/11681878_14

[16] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9, 3-4 (2014), 211–407. doi:10.1561/0400000042

[17] Huang Fang, Xiaoyun Li, Chenglin Fan, and Ping Li. 2023. Improved Convergence of Differential Private SGD with Gradient Clipping. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. https://openreview.net/forum?id=FRLswckPXQ5

[18] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. 2008. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, Cynthia Dwork (Ed.). ACM, 197–206. doi:10.1145/1374376.1374407

[19] Shafi Goldwasser, Michael P. Kim, Vinod Vaikuntanathan, and Or Zamir. 2022. Planting Undetectable Backdoors in Machine Learning Models : [Extended Abstract]. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*. IEEE, 931–942. doi:10.1109/FOCS54457.2022.00092

[20] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. 1985. The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract). In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, Robert Sedgewick (Ed.). ACM, 291–304. doi:10.1145/22145.22178

[21] Aparna Gupte, Neekon Vafa, and Vinod Vaikuntanathan. 2022. Continuous LWE is as Hard as LWE & Applications to Learning Gaussian Mixtures. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*. IEEE, 1162–1173. doi:10.1109/FOCS54457.2022.00112

[22] Andreas Haeberlen, Benjamin C. Pierce, and Arjun Narayan. 2011. Differential Privacy Under Fire. In *20th USENIX Security Symposium, San Francisco, CA, USA, August 8-12, 2011, Proceedings*. USENIX Association. http://static.usenix.org/events/sec11/tech/full_papers/Haeberlen.pdf

[23] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. 2010. Boosting the Accuracy of Differentially Private Histograms Through Consistency. *Proc. VLDB Endow.* 3, 1 (2010), 1021–1032. doi:10.14778/1920841.1920970

[24] Jamie Hayes, Borja Balle, and Saeed Mahloujifar. 2023. Bounding training data reconstruction in DP-SGD. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). http://papers.nips.cc/paper_files/paper/2023/hash/f8928b073ccbec15d35f2a9d39430bfd-Abstract-Conference.html

[25] Naoise Holohan, Stefano Braghin, Pól Mac Aonghusa, and Killian Levacher. 2019. Diffprivlib: The IBM Differential Privacy Library. *CoRR* abs/1907.02444 (2019). arXiv:1907.02444 http://arxiv.org/abs/1907.02444

[26] Naoise Holohan, Stefano Braghin, and Mohamed Suliman. 2024. Securing Floating-Point Arithmetic for Noise Addition. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*, Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie (Eds.). ACM, 1954–1966. doi:10.1145/3658644.3690347

[27] Nicholas J. Hopper, John Langford, and Luis von Ahn. 2002. Provably Secure Steganography. In *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings (Lecture Notes in Computer Science, Vol. 2442)*, Moti Yung (Ed.). Springer, 77–92. doi:10.1007/3-540-45708-9_6

[28] Christina Ilvento. 2020. Implementing the Exponential Mechanism with Base-2 Differential Privacy. In *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna (Eds.). ACM, 717–742. doi:10.1145/3372297.3417269

[29] Jiankai Jin, Eleanor McMurtry, Benjamin I. P. Rubinstein, and Olga Ohrimenko. 2022. Are We There Yet? Timing and Floating-Point Attacks on Differential Privacy Systems. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*. IEEE, 473–488. doi:10.1109/SP46214.2022.9833672

[30] Peter Kairouz, Ziyu Liu, and Thomas Steinke. 2021. The Distributed Discrete Gaussian Mechanism for Federated Learning with Secure Aggregation. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 5201–5212. http://proceedings.mlr.press/v139/kairouz21a.html

[31] Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. 2021. Preventing Manipulation Attack in Local Differential Privacy Using Verifiable Randomization Mechanism. In *Data and Applications Security and Privacy XXXV - 35th Annual IFIP WG 11.3 Conference, DBSec 2021, Calgary, Canada, July 19-20, 2021, Proceedings (Lecture Notes in Computer Science, Vol. 12840)*, Ken Barker and Kambiz Ghazinour (Eds.). Springer, 43–60. doi:10.1007/978-3-030-81242-3_3

[32] Dana Keeler, Chelsea Komlo, Emily Lepert, Shannon Veitch, and Xi He. 2023. DPrio: Efficient Differential Privacy with High Utility for Prio. *Proc. Priv. Enhancing Technol.* 2023, 3 (2023), 375–390. doi:10.56553/POPETS-2023-0086

[33] Dong-U Lee, John D. Villasenor, Wayne Luk, and Philip Heng Wai Leong. 2006. A Hardware Gaussian Noise Generator Using the Box-Muller Method and Its Error Analysis. *IEEE Trans. Computers* 55, 6 (2006), 659–671. doi:10.1109/TC.2006.81

[34] Haoran Li, Li Xiong, Xiaoqian Jiang, and Jinfei Liu. 2015. Differentially Private Histogram Publication for Dynamic Datasets: an Adaptive Sampling Approach. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis, and Jeffrey Xu Yu (Eds.). ACM, 1001–1010. doi:10.1145/2806416.2806441

[35] Johan Lokna, Anouk Paradis, Dimitar I. Dimitrov, and Martin T. Vechev. 2023. Group and Attack: Auditing Differential Privacy. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023*, Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda (Eds.). ACM, 1905–1918. doi:10.1145/3576915.3616607

[36] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. 2010. On Ideal Lattices and Learning with Errors over Rings. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings (Lecture Notes in Computer Science, Vol. 6110)*, Henri Gilbert (Ed.). Springer, 1–23. doi:10.1007/978-3-642-13190-5_1

[37] G. Marsaglia and T. A. Bray. 1964. A Convenient Method for Generating Normal Variables. *SIAM Rev.* 6, 3 (1964), 260–264. http://www.jstor.org/stable/2027592

[38] George Marsaglia and Wai Wan Tsang. 2000. The Ziggurat Method for Generating Random Variables. *Journal of Statistical Software* 5, 8 (2000), 1–7. doi:10.18637/jss.v005.i08

[39] Ilya Mironov. 2012. On significance of the least significant bits for differential privacy. In *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, Ting Yu, George Danezis, and Virgil D. Gligor (Eds.). ACM, 650–661. doi:10.1145/2382196.2382264

[40] Arjun Narayan, Ariel Feldman, Antonis Papadimitriou, and Andreas Haeberlen. 2015. Verifiable differential privacy. In *Proceedings of the Tenth European Conference on Computer Systems, EuroSys 2015, Bordeaux, France, April 21-24, 2015*, Laurent Réveillère, Tim Harris, and Maurice Herlihy (Eds.). ACM, 28:1–28:14. doi:10.1145/2741948.2741978

[41] Joseph P. Near, David Darais, Chike Abuah, Tim Stevens, Pranav Gaddamadugu, Lun Wang, Neel Somani, Mu Zhang, Nikhil Sharma, Alex Shan, and Dawn Song. 2019. Duet: An Expressive Higher-order Language and Linear Type System for Statically Enforcing Differential Privacy. *CoRR* abs/1909.02481 (2019). arXiv:1909.02481 http://arxiv.org/abs/1909.02481

[42] Chris Peikert. 2009. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, Michael Mitzenmacher (Ed.). ACM, 333–342. doi:10.1145/1536414.1536461

[43] Chris Peikert and Brent Waters. 2008. Lossy trapdoor functions and their applications. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, Cynthia Dwork (Ed.). ACM, 187–196. doi:10.1145/1374376.1374406

[44] Krishna Pillutla, Galen Andrew, Peter Kairouz, H. Brendan McMahan, Alina Oprea, and Sewoong Oh. 2023. Unleashing the Power of Randomization in Auditing Differentially Private ML. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). http://papers.nips.cc/paper_files/paper/2023/hash/d09ef5264966e17adffd3157265c9946-Abstract-Conference.html

[45] Ali Rahimi and Benjamin Recht. 2007. Random Features for Large-Scale Kernel Machines. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis (Eds.). Curran Associates, Inc., 1177–1184. https://proceedings.neurips.cc/paper/2007/hash/013a006f03dbc5392effeb8f18fda755-Abstract.html

[46] Zachary Ratliff and Salil P. Vadhan. 2024. A Framework for Differential Privacy Against Timing Attacks. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA,*

October 14-18, 2024, Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie (Eds.). ACM, 3615–3629. doi:10.1145/3658644.3690206

[47] Jason Reed and Benjamin C. Pierce. 2010. Distance makes the types grow stronger: a calculus for differential privacy. In *Proceedings of the 15th ACM SIGPLAN International Conference on Functional Programming, ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010*, Paul Hudak and Stephanie Weirich (Eds.). ACM, 157–168. doi:10.1145/1863543.1863568

[48] Oded Regev. 2005. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, Harold N. Gabow and Ronald Fagin (Eds.). ACM, 84–93. doi:10.1145/1060590.1060603

[49] Oded Regev. 2009. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM* 56, 6 (2009), 34:1–34:40. doi:10.1145/1568318.1568324

[50] Roie Reshef, Anan Kabaha, Olga Seleznova, and Dana Drachsler-Cohen. 2024. Verification of Neural Networks' Local Differential Classification Privacy. In *Verification, Model Checking, and Abstract Interpretation - 25th International Conference, VMCAI 2024, London, United Kingdom, January 15-16, 2024, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 14500)*, Rayna Dimitrova, Ori Lahav, and Sebastian Wolff (Eds.). Springer, 98–123. doi:10.1007/978-3-031-50521-8_5

[51] Alessio Di Santo. 2025. Lazarus Group Targets Crypto-Wallets and Financial Data while employing new Tradecrafts. *CoRR* abs/2505.21725 (2025). arXiv:2505.21725 doi:10.48550/ARXIV.2505.21725

[52] Kulin Shah, Sitan Chen, and Adam R. Klivans. 2023. Learning Mixtures of Gaussians Using the DDPM Objective. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). http://papers.nips.cc/paper_files/paper/2023/hash/3ec077b4af90f2556b517b556e186f64-Abstract-Conference.html

[53] Ali Shahin Shamsabadi, Gefei Tan, Tudor Cebere, Aurélien Bellet, Hamed Haddadi, Nicolas Papernot, Xiao Wang, and Adrian Weller. 2024. Confidential-DPproof: Confidential Proof of Differentially Private Training. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net. https://openreview.net/forum?id=PQY2v6VtGe

[54] Michael Shoemate, Andrew Vyrros, Chuck McCallum, Raman Prasad, Philip Durbin, Sílvia Casacuberta Puig, Ethan Cowan, Vicki Xu, Zachary Ratliff, Nicolás Berrios, Alex Whitworth, Michael Eliot, Christian Lebeda, Oren Renard, and Claire McKay Bowen. [n. d.]. OpenDP Library. https://github.com/opendp/opendp

[55] Haochen Sun and Xi He. 2025. VDDP: Verifiable Distributed Differential Privacy under the Client-Server-Verifier Setup. *CoRR* abs/2504.21752 (2025). arXiv:2504.21752 doi:10.48550/ARXIV.2504.21752

[56] 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. *CoRR* abs/1709.02753 (2017). arXiv:1709.02753 http://arxiv.org/abs/1709.02753

[57] Justin Thaler. 2022. Proofs, Arguments, and Zero-Knowledge. *Found. Trends Priv. Secur.* 4, 2-4 (2022), 117–660. doi:10.1561/3300000030

[58] Anvith Thudi, Hengrui Jia, Casey Meehan, Ilia Shumailov, and Nicolas Papernot. 2024. Gradients Look Alike: Sensitivity is Often Overestimated in DP-SGD. In *33rd USENIX Security Symposium, USENIX Security 2024, Philadelphia, PA, USA, August 14-16, 2024*, Davide Balzarotti and Wenyuan Xu (Eds.). USENIX Association. https://www.usenix.org/conference/usenixsecurity24/presentation/thudi

[59] Yuxin Wang, Zeyu Ding, Daniel Kifer, and Danfeng Zhang. 2020. CheckDP: An Automated and Integrated Approach for Proving Differential Privacy or Finding Precise Counterexamples. In *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna (Eds.). ACM, 919–938. doi:10.1145/3372297.3417282

[60] Jia Xu, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, Ge Yu, and Marianne Winslett. 2013. Differentially private histogram publication. *VLDB J.* 22, 6 (2013), 797–822. doi:10.1007/S00778-013-0309-Y

[61] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. 2021. Opacus: User-Friendly Differential Privacy Library in PyTorch. *CoRR* abs/2109.12298 (2021). arXiv:2109.12298 https://arxiv.org/abs/2109.12298

Received October 2025; revised January 2026; accepted February 2026