Zero Knowledge Differential Privacy

COMPSCI 2080: Applied Privacy for Data Science

Emily Kang, Sol Kim, Jaray Liu, Max Wagner, Weiyuan Gong

Abstract

Differential Privacy (DP) has become a widely adopted standard for the release of aggregate statistics over sensitive data. However, DP's reliance on randomness introduces a subtle risk: a malicious data curator could intentionally manipulate results while blaming discrepancies on noise. To address this issue, Biswas and Cormode [1] proposed a theoretical framework of verifiable DP, where the data curator must produce a zero-knowledge proof certifying that the reported output is both differentially private and correctly computed. This ensures verifiability without revealing the private randomness or compromising privacy. Nevertheless, the instantiation of their verifiable DP counting queries relies heavily on homomorphic commitments and unbiased public coin flipping to achieve DP using unbiased binomial mechanism. In this project, we propose a similar protocol to achieve a biased binomial mechanism. To this means, we construct a verifiable biased public coin flipping protocol. We also further propose a zero-knowledge randomized response protocol satisfying verifiable DP, which does not rely on the homomorphic property of commitments. Finally, to demonstrate zero-knowledge to a broader audience, we implement an interactive interface in which a malicious prover can attempt to cheat throughout the protocol for the unbiased binomial mechanism.

Motivations and Problem Statement

Our project is motivated by the growing need for verifiable privacy in settings where the assumption of a trusted central data curator no longer holds. While Differential Privacy (DP) provides strong formal guarantees about the protection of individual data, these guarantees critically depend on the honest implementation of the DP mechanism by the curator. In reality, **a malicious curator could alter true counts or manipulate the added noise while falsely claiming to adhere to a DP protocol**—undermining the integrity of the system and violating user trust. This fundamental flaw motivated our exploration of Zero Knowledge Differential Privacy (ZKDP), a cryptographic approach where the curator must prove to an external verifier that the DP mechanism was applied correctly, without revealing any sensitive underlying data or the precise noise used. Although none of us had prior experience with cryptography, we were excited to dive into this intersection, especially after encountering the paper **Interactive Proofs for Differentially Private Counting by Biswas and Cormode (2023)**, which presents a framework for adding Binomial noise to counts in a verifiable way. Given the novelty of the ZKDP literature, we saw a valuable opportunity to build on this work—both adapting their original codebase into a web-based interface similar to DP Wizard and formulating our own ZKDP proofs for other canonical DP mechanisms. With this, we hope to encourage broader adoption and further development at the intersection of cryptography and privacy, addressing the fundamental limitation in the trust model of DP.

Contact

Emily Kang: emily_kang@college.harvard.edu
Sol Kim: solkim@college.harvard.edu
Jaray Liu: jarayliu@college.harvard.edu
Max Wagner: maxwagner@college.harvard.edu
Weiyuan Gong: wgong@g.harvard.edu

Verifiable Binomial Mechanism

A zero-knowledge proof (ZKP) is an interactive protocol between a prover P and a verifier V by which P **convinces V that a statement x is true without revealing any additional information.**

Such a protocol π satisfies three properties: **completeness** (an honest prover convinces an honest verifier except with negligible probability), **soundness** (no cheating prover can convince the verifier of a false statement except with negligible probability), and **zero-knowledge** (there exists a simulator that, given only the statement x, can produce a transcript indistinguishable from a real execution with witness w).

Differential privacy (DP) protects sensitive data by adding calibrated noise to query outputs. For a dataset $X \in Z_q^n$ and the counting query $Q(X) = \sum_{i=1}^n x_i$, **the binomial mechanism** samples noise $Z \sim \text{Binomial}(n_b, p)$ and releases $\mathcal{M}(X) = Q(X) + Z$.

This mechanism satisfies (ε, δ)-differential privacy with $\varepsilon = 10\sqrt{\frac{\ln(2/\delta)}{n_b\, p\,(1-p)}}, \quad \delta = o\left(\frac{1}{n_b}\right)$.

Results and Discussion

Our first contribution is a **publicly verifiable biased coin-flipping protocol** that enables a prover to simulate a Bernoulli random variable with arbitrary bias p= k/m, where k and m are integers. To do so, the prover commits to an m-bit vector containing exactly k 1's, using Pedersen commitments to preserve both binding and hiding properties. A single fair public coin is then used to select an index uniformly at random; the bit revealed at this position constitutes the Bernoulli(p) draw. To ensure correctness, the verifier checks that the commitments encode valid bits and that their product opens to the declared total k. This protocol guarantees that the sampled bit has the correct marginal distribution as long as either the prover or the verifier is honest, but assumes both parties follow the commitment setup faithfully. While this method incurs a communication overhead of O(m) per draw—since m commitments must be shared and validated—it offers a simple way to simulate biased randomness in a verifiable manner for more complex mechanisms.

Biased bit flipping algorithm. To begin with, we use a fair coin flipping oracle [1] to simulate the noise of one Bern(p) trial for general $p \in \mathbb{Q}$ in Algorithm 1.

```
Algorithm 1: Biased bit flipping.
  Input:  Biased probability $p$.
  Output: A biased random coin.
  1: Publicly represent $p$ as a rational number $\frac{k}{m}$. $k$ and $m$ are publicly known.
  2: The prover allocates a $m$ bit vector $(v_1, \dots, v_m)$ and randomly samples $k$ of them. Set those $k$ bits
     equal to 1 and the rest equal to 0.
  3: The prover generates Pedersen commitments for the $m$ bits $C_i = \mathrm{Com}(v_i; r_i)$, and sends all $C_i$ to the
     verifier. Note that these values are subject to the binding property of Pedersen commitments.
  4: The verifier uses Σ-OR [6] on each of the $C_i$ to verify that each bit is either 0 or 1.
  5: The verifier computes $C_{\mathrm{sum}} := \prod_{i=1}^m C_i$ using the $m$ commitments received. The prover reveals the
     aggregate randomness $r_\Sigma := \sum_i r_i$ which the verifier uses with publicly known $k$ to compute
     $\mathrm{Com}(k; r_\Sigma)$. The verifier accepts if and only if $C_{\mathrm{sum}} = \mathrm{Com}(k; r_\Sigma)$.
  6: The verifier randomly samples an index $j \in \{1, .., m\}$. The bit at that index is 1 with probability
     $\frac{k}{m} = p$, and 0 with probability $1 - p$ because it was sampled uniformly at random.
  7: The verifier already has the commitment of the bit at the $j$-th index, which they can use to do other
     checks.
```

Building on this biased coin-flipping protocol, we construct a **verifiable protocol for the biased Binomial mechanism**, which releases Q(X)+Z, where Q(X) is a counting query and $Z \sim \text{Binomial}(n_b, p)$. By repeating the biased coin-flipping protocol n_b times, we obtain n_b Bernoulli samples with bias p, and their sum constitutes the desired noise.

The prover commits to the original input data and the sampled noise bits, then releases the noisy output alongside a commitment that aggregates both contributions. The verifier checks that the sum of the commitments matches the released value, leveraging the homomorphic property of Pedersen commitments. This mechanism preserves differential privacy while offering cryptographic guarantees that the noise was faithfully generated

The main limitations here lie in scalability: the communication overhead grows with both m and n_b, and any break in the discrete log assumption underlying the commitment scheme would compromise security. Still, our theoretical result shows that **verifiable DP for biased Binomial noise is achievable under minimal assumptions**, a significant generalization of prior work. Note that this algorithm is an extension of of the unbiased Binomial mechanism in [1].

```
Algorithm 2: Verifiable differential private counting based on biased binomial mechanism
  1: Vfr and Pv agree on the shared parameters. For the Pedersen commitments, these are $G_q$, $g$, and $h$.
  2: Pv sends the commitments of each $x_i$ with their own randomness as $\mathrm{Com}(x_i, r_i)$ to Vfr.
  3: Using a Σ-OR protocol, we check that each $x_i$ is a bit.
  4: Pv and Vfr sample the biased coins using Algorithm 1. Each time Algorithm 1 is run, the prover
     knows the value of the chosen bit $v \in \{0,1\}$ that was equal to 1 with probability $p$ for rational $p$,
     which they initially committed as $c_v = \mathrm{Com}(v, s)$. Meanwhile, the verifier has the commitment of
     this bit $v$, $c_v$. Algorithm 1 is run $n_b$ times; the Binomial noise comes from summing all $v$'s: $\sum_{j=1}^{n_b} v_j$.
     The commitment of this sum can be calculated by the verifier as $\prod_{j=1}^{n_b} c_{v_j}$.
  5: Pv releases the raw noisy count $x_i + \sum_{j=1}^{n_b} v_j$, as well as the sum of all the noise added to the
     Pedersen commitments: $\sum_{i=1}^n r_i + \sum_{j=1}^{n_b} s_j$, where $\sum_{j=1}^{n_b} s_j$ is the sum of the noise used to commit the
     chosen $v_j$ from each of the $n_b$ runs of Algorithm 1.
  6: The remaining part follows the same steps as lines 12–13 in Figure 1.
```

Our final contribution is a **zero-knowledge protocol for randomized response that can potentially avoid homomorphic commitments**. Instead, the prover generates k private fair coins, which are XORed with k public fair coins from the Morra oracle to create a uniform random number U in the range {0, ..., 2^k - 1}. The prover then flips their bit x with probability p = m / 2^k by computing the output o = x XOR [U < m]. All values involved are committed to using standard one-way commitments, and the prover provides a zero-knowledge proof that the output was computed correctly from the committed values. This approach significantly reduces cryptographic overhead and removes the need for group structure, relying only on the existence of one-way functions. While the current design is limited to binary outputs and flip probabilities where p is a rational number with denominator a power of two, it demonstrates that verifiable differential privacy can be achieved under minimal cryptographic assumptions.

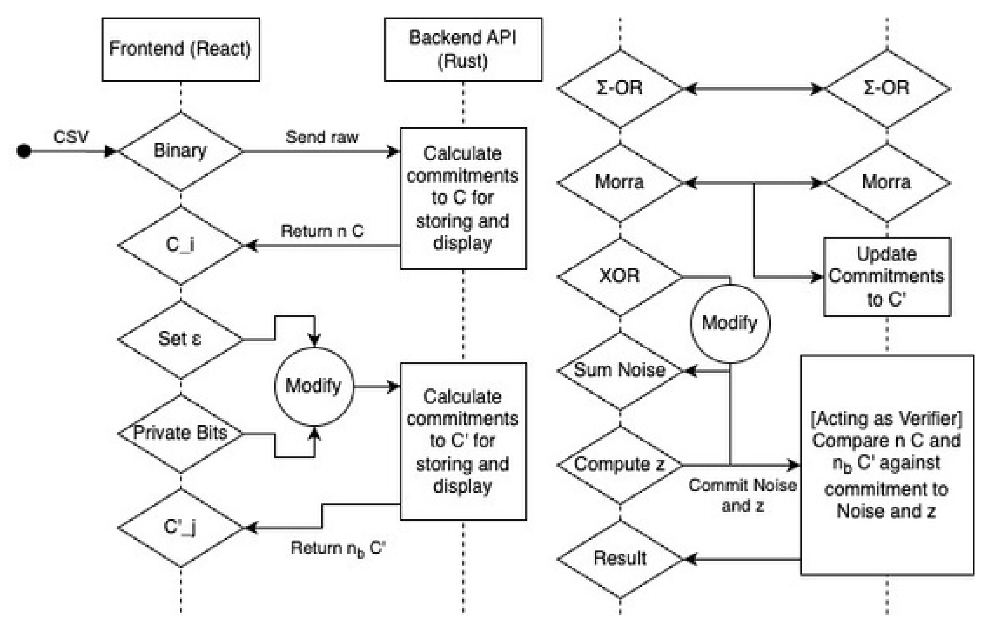
```
Algorithm 3: Alternative Randomised response with flip probability $p$
  Input:  A rational probability $p = m/2^k$, such that $m$ and $k$ are integers; client's private data $x \in \{0,1\}$
  Output: A single biased bit $o$ revealed to the verifier
  1: Prover commits $C_x, C_{v_1}, \dots, C_{v_k}$ and gives a Σ-OR proof each commitment is a bit.
  2: Generate $b_1, \dots, b_k \leftarrow \mathcal{O}_{\mathrm{Morra}}$.
  3: Prover computes $d_i := v_i \oplus b_i$ and $U = \sum_{i=1}^k d_i 2^{i-1}$.
  4: Output $o := x \oplus [U < m]$.
  5: Prover proves in zero knowledge that (i) Line 1 was honest and (ii) Line 4 holds for the committed
     values.
```

Website Demo

Alongside our theoretical advances in zero-knowledge protocols for randomized response, we built a fully interactive web application—hosted at **zkdp.org**—that brings the element of intractability and public usage to our project. At its core sits a Rust backend, deployed on Railway, which contains the entire verifiable binomial mechanism—based on the existing work done by Ari Biswas. This server tracks all Pedersen commitments (using the Ristretto construction over a prime-order elliptic curve), orchestrates Σ-OR proofs to show each committed value is a valid bit, runs the Morra coin-flipping protocol to generate public randomness, and finally verifies the prover's claims on both their raw data and added noise before returning a cryptographically certified noisy count. Each phase of the protocol—data commitment, bit-proof submission, randomness exchange, and proof aggregation—is exposed via simple JSON GET/POST endpoints, making the flow easy to integrate with any client.

The React frontend, written in TypeScript, guides users step-by-step through the protocol. After uploading a CSV file and choosing a column and threshold, the interface prompts users to commit their data values; it then invites them to play the Morra game against the server to generate provably fair random bits. As each commitment or proof is submitted, the site visualizes the underlying zero-knowledge transcript, and if a user attempts to "cheat" by altering a bit or misreporting randomness, the Rust verifier immediately flags the inconsistency. The UI then informs the user that the check failed, teaching users why zero-knowledge proofs enforce both privacy and correctness. All code is open-source on GitHub (**github.com/mwagner6/zkdp-exponential**), and future updates will extend the site to support biased-binomial and randomized-response mechanisms, port critical libraries to WebAssembly, and integrate with the OpenDP ecosystem for seamless adoption of verifiable DP.

Website API Flow



References

[1] Biswas, A., & Cormode, G. (2023). Interactive proofs for differentially private counting. Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, 1919–1933.
[2] Gaboardi, M., Hay, M., & Vadhan, S. (2020). A programming framework for OpenDP. Manuscript, May.
[3] Pedersen, T. P. (1991). Non-interactive and information-theoretic secure verifiable secret sharing. In Annual International Cryptology Conference (pp. 129–140). Springer.
[4] Vadhan, S. (2017). The complexity of differential privacy. In Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich (pp. 347–450). Springer.
[5] Mironov, I., Pandey, O., Reingold, O., & Vadhan, S. (2009). Computational differential privacy. In Annual International Cryptology Conference (pp. 126–142). Springer.
[6] Ghazi, B., Golowich, N., Kumar, R., Pagh, R., & Velingker, A. (2021). On the power of multiple anonymous messages: Frequency estimation and selection in the shuffle model of differential privacy. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 463–488). Springer.
[7] Cramer, R., Damgård, I., & Schoenmakers, B. (1994). Proofs of partial knowledge and simplified design of witness hiding protocols. In Annual International Cryptology Conference (pp. 174–187). Springer.
[8] Knuth, D. E. (1976). The complexity of nonuniform random number generation. In Algorithm and Complexity: New Directions and Results. Academic Press.