Cryptϵ: Crypto-Assisted Differential Privacy on Untrusted Servers

Amrita Roy Chowdhury
University of Wisconsin-Madison
amrita@cs.wisc.edu

Chenghong Wang
Duke University
cw374@duke.edu

Xi He
University of Waterloo
xihe@uwaterloo.ca

Ashwin Machanavajjhala
Duke University
ashwin@cs.duke.edu

Somesh Jha
University of Wisconsin-Madison
jha@cs.wisc.edu

Abstract

In this work, we propose, Cryptϵ, a system and programming framework that (1) achieves the accuracy guarantees and algorithmic expressibility of the central model (2) without any trusted data collector like in the local model. Cryptϵ achieves the "best of both worlds" by employing two non-colluding untrusted servers that run DP programs on encrypted data from the data owners. Although straightforward implementations of DP programs using secure computation tools can achieve the above goal theoretically, in practice they are beset with many challenges such as poor performance and tricky security proofs. To this end, Cryptϵ allows data analysts to author logical DP programs that are automatically translated to secure protocols that work on encrypted data. These protocols ensure that the untrusted servers learn nothing more than the noisy outputs, thereby guaranteeing ϵ-DP for all Cryptϵ programs. Cryptϵ supports a rich class of DP programs that can be expressed via a small set of transformation and measurement operators followed by arbitrary post-processing. Further, we propose performance optimizations leveraging the fact that the output is noisy. We demonstrate Cryptϵ's feasibility for practical DP analysis with extensive empirical evaluations on real datasets.

1 Introduction

Differential privacy (DP) is a rigorous privacy definition that has become the gold standard for data privacy. It is typically implemented in one of two models – *centralized differential privacy* (CDP) and *local differential privacy* (LDP). In CDP, data from individuals are collected and stored *in the clear* in a *trusted* centralized data curator which then executes DP programs on the sensitive data and releases outputs to an untrustworthy data analyst. In LDP, there is no trusted data curator. Rather, each individual perturbs his/her own data using a (local) DP algorithm. The data analyst uses these noisy data to infer aggregate statistics of the datasets. In practice, CDP's assumption of a trusted server is ill-suited for many applications as it constitutes a single point of failure for data breaches, and saddles the trusted curator with legal and ethical obligations to uphold data privacy. Hence recent commercial deployments of DP [14, 18] have preferred LDP over CDP. However, LDP's attractive privacy properties comes at a cost. Under the CDP model, the expected additive error for a aggregate count over a dataset of size n is at most $\Theta(1/\epsilon)$ to achieve ϵ-DP. In contrast, under the LDP model, at least $\Omega(\sqrt{n}/\epsilon)$ additive expected error must be incurred by any ϵ-DP program [6, 9, 11], owing to the randomness of each data owner. The LDP model in fact imposes additional penalties on the algorithmic expressibility; the power of LDP is equivalent to that of the statistical query model [22] and there exists an exponential separation between the accuracy and sample complexity of LDP and CDP algorithms [21].

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

In this paper, we strive to bridge the gap between LDP and CDP. We propose, Cryptϵ, a system and a programming framework for executing DP programs that:

- never stores or computes on sensitive data in the clear
- achieves the accuracy guarantees and algorithmic expressibility of the CDP model

Cryptϵ employs a pair of untrusted but non-colluding servers – Analytics Server (AS) and Cryptographic Service Provider (CSP). The AS executes DP programs (like the data curator in CDP) but on *encrypted* data records. The CSP initializes and manages the cryptographic primitives, and collaborates with the AS to generate the program outputs. Under the assumption that the AS and the CSP are semi-honest and do not collude (a common assumption in cryptographic systems [28, 27, 16, 23, 26, 17, 15]), Cryptϵ ensures ϵ-DP guarantee for its programs via two cryptographic primitives – linear homomorphic encryption (LHE) and garbled circuits.

Cryptϵ provides a data analyst with a programming framework to author logical DP programs just like in CDP. Like in prior work [25, 13, 29], access to the sensitive data is restricted via a set of predefined transformations operators (inspired by relational algebra) and DP measurement operators (Laplace mechanism and Noisy-Max [12]). Thus, any program that can be expressed as a composition of the above operators automatically satisfies ϵ-DP (in CDP model) giving the analyst a proof of privacy for free. Cryptϵ programs support constructs like looping, conditionals, and can arbitrarily post-process outputs of measurement operators.
The main contributions of this work are:

- **New Approach**: We present the design and implementation of Cryptϵ, a novel system and programming framework for executing DP programs over encrypted data on two non-colluding untrusted servers.
- **Algorithm Expressibility**: Cryptϵ supports a rich class of state-of -the-art DP programs expressed in terms of a small set of transformation and measurement operators. Thus, Cryptϵ achieves the accuracy guarantees of the CDP model without the need for a trusted curator.
- **Ease Of Use:** Cryptϵ lets data analysts express the DP program logic using high level operators. Cryptϵ automatically translates this to the underlying implementation specific secure protocols that work on encrypted data and provides a DP guarantee (in the CDP model) for free. Thus the data analyst is relieved of all concerns regarding implementation of secure computation protocols.
- **Performance Optimizations**: We propose optimizations that speed up computation on encrypted data by at least an order of magnitude. A novel contribution of this work is a DP indexing optimization that leverages the fact that intermediate statistics about the data can be revealed as long as DP is satisfied.
- **Practical for Real World Usage**: For the same tasks, Cryptϵ programs achieve accuracy comparable to CDP and at least 2 orders of magnitude more accurate than that of LDP. Cryptϵ runs within 5 min for a large class of programs on a dataset with 32,561 rows and 4 attributes.
- **Generalized Multiplication Using LHE**: Our implementation uses an efficient way for performing n-way multiplications using LHE which maybe of independent interest.

The full version of the paper is available at the link [1].

2 Cryptϵ Overview

2.1 System Architecture

Figure 1 shows Cryptϵ's system architecture. Cryptϵ has two servers: Analytics server (AS) and Cryptographic Service Provider (CSP). At the very outset, the CSP records the total privacy budget, ϵ^B, (provided by the data owners) and generates the key pair (pk (public key), sk (secret key)) for the encryption scheme. The data owners, $\mathsf{DO}_i, i \in [m]$ (m = number of data owners), encrypt their data records, D_i, in the appropriate format with the public key (pk) and send the encrypted records, $\tilde{\boldsymbol{D}}_{\boldsymbol{i}}$, to the AS which aggregates them into a single encrypted database $\tilde{\boldsymbol{\mathcal{D}}}$. Next, the AS inputs logical programs from the data analysts and translates them to Cryptϵ's implementation specific secure protocols that work on $\tilde{\boldsymbol{\mathcal{D}}}$. A Crypt$\epsilon$ program typically consists of a sequence of transformation operators followed by a measurement operator. The AS can execute most of the transformations on its own. However, each measurement operator requires an interaction with the CSP for (a) decrypting

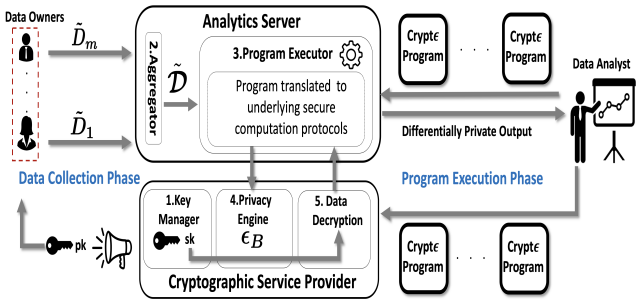

Figure 1: Cryptϵ System: AS executes Cryptϵ programs; CSP manages the cryptographic primitives.

the answer, and (b) checking that the total privacy budget, ϵ^B, is not exceeded. In this way, the AS and the CSP compute the output of a Cryptϵ program with the data owners being offline.

2.2 Cryptϵ Design Principles

Minimal Trust Assumptions: In order to accommodate the use of cryptographic primitives, we assume a computationally bounded adversary in Cryptϵ. However, a generic $m+1$ party SMC would be computationally expensive. This necessitates a third party entity that can capture the requisite secure computation functionality in (at least) a 2-party protocol instead. For this two-server model, we assume semi-honest behaviour and non-collusion. This is a very common assumption in the two-server model [28, 27, 16, 23, 26, 17, 15].

Programming Framework: Conceptually, the aforementioned goal of achieving the *best of both worlds* can be obtained by implementing the required DP program using off-the-self secure multi-party computation (SMC) tools like [2, 5, 4, 3]. However, when it comes to real world usage, Cryptϵ outperforms such approaches due to the following reasons.

First, without the support of a programming framework like that of Cryptϵ, every DP program must be implemented from scratch. This requires the data analyst to be well versed in both DP and SMC techniques; he/she must know how to correctly manage keys, implement SMC protocols, estimate sensitivity of transformations and track privacy budget across programs. In contrast, Cryptϵ allows data analysts to write the DP program using a high-level and expressive programming framework. Cryptϵ abstracts out all the low-level implementation details like the choice of input data format, translation of queries to that format, choice of SMC primitives and privacy budget monitoring from the analyst thereby reducing his/her burden of complex decision making. Thus every Cryptϵ program is automatically translated to protocols corresponding to the underlying implementation.

Second, SMC protocols can be prohibitively costly in practice unless they are carefully tuned to the application. Cryptϵ supports optimized implementations for a small set of operators, which results in efficiency for all Cryptϵ programs.

Third, a DP program can be typically divided into segments that (i) transform the private data, (ii) perform noisy measurements, and (iii) post-process the noisy measurements without touching the private data. A naive implementation may implement all the steps using SMC protocols even though post-processing can be performed in the clear. Given a DP program written in a general purpose programming language (like Python), automatically figuring out what can be done in the clear can be subtle. In Cryptϵ programs, however, transformation, measurement are clearly delineated, as the data can be accessed only through a prespecified set of operators. Thus, SMC protocols are only used for transformation and measurement operations, which improves performance.

Last, the security (privacy) proofs for just stand-alone cryptographic and DP mechanisms can be notoriously tricky [7, 24]. Combining the two thus exacerbates the technical complexity, making the design vulnerable to faulty proofs [19].

Data Owners are Offline: Recall, Cryptϵ's goal is to mimic the CDP model with untrusted servers. Hence, it is designed so that the data owners are offline after submitting their encrypted records to the AS. If the data owners were online, the efficiency of some programs could be improved as some of the computation can be offloaded to the data owners.

Low burden on CSP: Cryptϵ views the AS as an extension of the analyst; the AS has a vested interest in obtaining the result of the programs. Thus we require the AS to perform the majority of the work for any Cryptϵ program execution; interactions with the CSP should be minimal and only related to data decryption. Keeping this in mind, we design the AS to perform most of the data transformations by itself. Specifically for every Cryptϵ program, the AS processes the whole database and transforms it into concise representations (like an encrypted scalar or a short vector) which is then decrypted by the CSP. An example real world setting can be when Google and Symantec assumes the role of the AS and the CSP respectively.

Separation of logical programming framework and underlying physical implementation: The programming framework is independent from the underlying implementation. This allows certain flexibility in the choice for the implementation. For example, our prototype Cryptϵ uses per attribute one-hot-encoding as the input data format. However, any other encoding scheme like multi-attribute one-hot-encoding, range based encoding can be used instead. As discussed above, due to our design choice of having low burden on the CSP, we implement Cryptϵ using LHE and garbled circuits. It is straightforward to replace LHE with the optimized HE scheme in [8] or garbled circuits with the mixed protocol ABY framework [10].

Yet another alternative implementation for Cryptϵ could be where the private database is equally shared between the two servers and they engage in a secret share based SMC protocol for executing the DP programs. This would require both the servers to do almost equal amount of work for each program. Such an implementation would be justified only if both the servers are equally invested in learning the DP statistics and is ill-suited for our context. A real world analogy for this can be if Google and Baidu decide to compute some statistics on their combined user bases.

2.3 Cryptϵ Optimization: Differentially Private Index Optimization

In this section we provide a brief overview of the DP index optimization which we consider to be a novel contribution of this paper. Additionally, we propose three other crypto-engineering optimizations for Cryptϵ which are detailed in the full paper [1].
The DP index optimization is motivated by the fact that several programs first filter (selection operator) out a large number of rows in the dataset. Since the database is encrypted, the naive filter implementation keeps all the rows (even if they do not satisfy the condition) as the AS has no way of telling whether the filter condition is satisfied. However, if there were an index on the filtering attribute, then the program can be executed only on the correct subset of row; but an exact index would violate DP. Hence, we propose a DP index to bound the information leakage while improving the performance. At a high level, the DP index on any ordinal attribute A is constructed as follows: (a) securely sort [20] the input encrypted database $\tilde{\boldsymbol{\mathcal{D}}}$ on A and (b) learn a mapping $\mathcal{F}$ from the domain of A to $[1, |\tilde{\boldsymbol{\mathcal{D}}}|]$ such that most of the rows with index less than $\mathcal{F}(v), v \in domain(A)$ have a value less than v for A. Cryptϵ learns this mapping under DP (details in [1]). When a Cryptϵ program starts with a filter $\phi = A \in [v_s, v_e]$, Cryptϵ derives the indices for a noisy subset of rows from $\mathcal{F}$ that satisfies the condition ϕ and executes the rest of the program on this subset.

3 Experimental Evaluation Highlights

We present the highlights of the experimental evaluation of Cryptϵ in this section.

- Cryptϵ can achieve up to 2 orders of smaller error than the corresponding LDP implementation on a data of significant size ($\sim 30,00$).
- The optimizations in Cryptϵ can improve the performance of unoptimized Cryptϵ by up to $5667\times$.
- A large class of Cryptϵ programs execute within 5 mins for a dataset of size $\sim 30,000$ and it scales linearly with the dataset size. The AS performs majority of the work for most programs.

4 Conclusions

In this paper we have proposed a system and programming framework, Cryptϵ, for differential privacy that achieves the constant accuracy guarantee and algorithmic expressibility of CDP without any trusted server. This is achieved via two non-colluding servers with the assistance of cryptographic primitives, specifically LHE and garbled circuits.

References

[1] Full version of the paper. https://drive.google.com/file/d/1sYposoEdI4XpNzJqAQIg67dXqNXN6Loj/view?usp=sharing.

[2] https://github.com/emp-toolkit.

[3] https://github.com/encryptogroup/aby.

[4] https://github.com/kuleuven-cosic/scale-mamba.

[5] http://www.multipartycomputation.com/mpc-software.

[6] A. Beimel, K. Nissim, and E. Omri. Distributed private data analysis: Simultaneously solving how and what. In *Proceedings of the 28th Annual Conference on Cryptology: Advances in Cryptology*, CRYPTO 2008, pages 451–468, Berlin, Heidelberg, 2008. Springer-Verlag.

[7] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Advances in Cryptology - EUROCRYPT 2006*, pages 409–426, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[8] M. Blatt, A. Gusev, Y. Polyakov, K. Rohloff, and V. Vaikuntanathan. Optimized homomorphic encryption solution for secure genome-wide association studies. *IACR Cryptology ePrint Archive*, 2019:223, 2019.

[9] T.-H. H. Chan, E. Shi, and D. Song. Optimal lower bound for differentially private multi-party aggregation. In *Proceedings of the 20th Annual European Conference on Algorithms*, ESA'12, pages 277–288, Berlin, Heidelberg, 2012. Springer-Verlag.

[10] D. Demmler, T. Schneider, and M. Zohner. Aby - a framework for efficient mixed-protocol secure two-party computation. In *NDSS*, 2015.

[11] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 429–438, Oct 2013.

[12] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9:211–407, Aug. 2014.

[13] H. Ebadi and D. Sands. Featherweight pinq, 2015.

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

[15] A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, and D. Evans. Secure linear regression on vertically partitioned datasets. *IACR Cryptology ePrint Archive*, 2016:892, 2016.

[16] A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, and D. Evans. Privacy-preserving distributed linear regression on high-dimensional data. *PoPETs*, 2017:345–364, 2017.

[17] I. Giacomelli, S. Jha, M. Joye, C. D. Page, and K. Yoon. Privacy-preserving ridge regression with only linearly-homomorphic encryption. In B. Preneel and F. Vercauteren, editors, *Applied Cryptography and Network Security*, pages 243–261, Cham, 2018. Springer International Publishing.

[18] A. Greenberg. Apple's 'differential privacy' is about collecting your data—but not *your* data. *Wired*, Jun 13 2016.

[19] X. He, A. Machanavajjhala, C. Flynn, and D. Srivastava. Composing differential privacy and secure computation: A case study on scaling private record linkage. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1389–1406, New York, NY, USA, 2017. ACM.

[20] K. V. Jónsson, G. Kreitz, and M. Uddin. Secure multi-party sorting and applications. Cryptology ePrint Archive, Report 2011/122, 2011. https://eprint.iacr.org/2011/122.

[21] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 531–540, Oct 2008.

[22] M. Kearns. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, Nov. 1998.

[23] S. Kim, J. Kim, D. Koo, Y. Kim, H. Yoon, and J. Shin. Efficient privacy-preserving matrix factorization via fully homomorphic encryption: Extended abstract. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '16, pages 617–628, New York, NY, USA, 2016. ACM.

[24] M. Lyu, D. Su, and N. Li. Understanding the sparse vector technique for differential privacy. *PVLDB*, 10:637–648, 2017.

[25] F. D. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, SIGMOD '09, pages 19–30, New York, NY, USA, 2009. ACM.

[26] P. Mohassel and Y. Zhang. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 19–38, May 2017.

[27] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh. Privacy-preserving matrix factorization. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 801–812, New York, NY, USA, 2013. ACM.

[28] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *2013 IEEE Symposium on Security and Privacy*, pages 334–348, May 2013.

[29] D. Zhang, R. McKenna, I. Kotsogiannis, M. Hay, A. Machanavajjhala, and G. Miklau. EKTELO: A framework for defining differentially-private computations. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 115–130, 2018.