Practical Two-party Computational Differential Privacy with Active Security

Fredrik Meisingseth
Graz University of Technology
Graz, Austria
fredrik.meisingseth@iaik.tugraz.at

Christian Rechberger
Graz University of Technology
Graz, Austria
christian.rechberger@iaik.tugraz.at

Fabian Schmid
Graz University of Technology
Graz, Austria
fabian.schmid@iaik.tugraz.at

Abstract

In this work we revisit the problem of using general-purpose MPC schemes to emulate the trusted dataholder in differential privacy (DP), to achieve the same accuracy but without the need to trust one single dataholder. In particular, we consider the two-party model where two computational parties (or dataholders), each with their own dataset, wish to compute a canonical DP mechanism on their combined data and to do so with active security. We start by remarking that available definitions of computational DP (CDP) for protocols are somewhat ill-suited for such a use-case, due to them either poorly capturing some strong security guarantees commonly given by general-purpose MPC protocols, or having too strict requirements in the sense that they need significant adjustment in order to be satisfiable by using common DP and MPC techniques. With this in mind, we propose a new version of simulation-based CDP, called SIM*-CDP, and prove it to be stronger than the IND-CDP and SIM-CDP and incomparable to SIM$^+$-CDP. We demonstrate the usability of the SIM*-CDP definition by showing how to satisfy it by the use of an available distributed protocol for sampling truncated geometric noise. Further, we use the protocol to compute two-party inner-products with CDP and active security, and with accuracy equal to that of the central model, being the first to do so. Finally, we provide an open-sourced implementation and benchmark its practical performance. Our implementation generates a truncated geometric sample in between about 0.035 and 3.5 seconds (amortized), depending on network and parameter settings, comparing favourably to existing implementations.

Keywords

Differential privacy, Multiparty computation, UC-security, Noise sampling

1 Introduction

The study of differential privacy in various distributed settings has given rise to a plethora of new definitions of DP, such as DP in the *local model (LDP)* [48], the *shuffle model* [6, 16] and definitions with a computationally bounded adversary, giving guarantees of *computational DP (CDP)* [4, 26, 64]. Each of the definitions is subject to its own restrictions in the adversarial model and in the accuracy that can be achieved within them. For instance, it is well-studied that in LDP, which is a computationally efficient model with very few trust assumptions, one must in some settings add much more noise than the standard central model of statistical DP (SDP)[1] [4, 15, 48, 72]. One recurring idea is that one can use general-purpose *multiparty computation (MPC)* techniques to *emulate* a trusted central dataholder and thus one may get the same accuracy as is in the central model without having to trust a central computational party [16, 29]. The troubles in realising this idea, which we can call *generic emulation of the dataholder (GED)*, are firstly that one must accept the, potentially, large computational costs of MPC and secondly that it is not necessarily clear how one should define DP in this new distributed and computational setting. In order to avoid or reduce the computational costs of using MPC, up until now, most of the works in this area have opted for considering passive adversaries [4, 30, 68], only allowing aggregate functions [19, 49] and/or requiring honest majorities [26]. We focus on the case of two parties[2], active (static) corruptions, and require efficient protocols[3] for non-aggregate functionalities that achieve the same accuracy as in the central model. This work consists of two main parts. First, we consider existing definitions of CDP for the setting above, conclude that they leave some things to be wished for and we therefore propose an adjusted definition of CDP. In the second part, we implement an existing protocol for noise sampling [30], prove that it fulfills our new definition (but not some previous ones) and show that when augmented to use a mixed-circuit approach, it is efficient also in practice.

Definitions of CDP. In order to design practical protocols for GED, we want a DP notion that is directly compatible with security definitions of state-of-the-art MPC schemes and that allows the emulated dataholder to compute common SDP mechanisms. Since we consider the case of two parties and active corruptions, for which information-theoretic general-purpose MPC is impossible [18, 32, 37], we have to use CDP [4, 64]. Intuitively, compatibility with standard MPC security definitions might seem immediate, since the possibility of general-purpose MPC means that essentially any functionality can be securely realised as long as it is computable in strict polynomial time. The restriction to polynomial-time functionalities may look minor but we shall see that it causes quite some intricacies, especially since it means that only functionalities whose

Proceedings on Privacy Enhancing Technologies 2025(1), 341–360

https://doi.org/10.56553/popets-2025-0019

[1] Throughout this work we use 'DP' to refer to definitions that are both statistical (information-theoretic) and computational. When distinguishing between them we use 'SDP' and 'CDP' respectively.
[2] We consider both in the discussion about definition and in that of protocols only the case of two-party computation, although since all the tools we use are also applicable to settings with more parties (and all definitions can trivially be extended to those settings), we will continue to speak of MPC at times. At all times, the reader can suitably think of the special case of two parties whenever MPC is mentioned.
[3] In particular, we require that the protocols are computable in strict polynomial time in a finite computational model, as suggested in [2].

output distribution is of a certain type can be securely realised. Critically, fundamental SDP mechanisms such as the Laplace [28], geometric [36], Gaussian [29] and discrete Gaussian [13] mechanisms have output distributions which cannot be computed exactly in strict polynomial time.[4] This means that the CDP definition we use needs to allow either that the protocol does not exactly emulate the dataholder (imperfect correctness) or that the emulated dataholder does not exactly compute the SDP mechanism, or both.

In Section 3, we revisit the CDP definitions for two-party protocols by [64]. Since we will refer to it recurrently, let us call the paper [64] *MPRV*, after its authors. They are all applicable to the setting of active corruptions however we find that they all fit unnaturally to the task of GED. For IND-CDP (Definition 3.1) and SIM-CDP (Definition 3.2), the inconvenience lies in that by using MPC to compute an SDP mechanism, one gets a protocol with much stronger guarantees than is captured in those CDP definitions, such as guarantees of security and correctness. This creates the need to analyse the desired properties of the protocol (now correctness, accuracy, security and CDP) separately, contrary to the custom when it comes to general-purpose MPC, which is typically analysed in the ideal/real paradigm where all such properties are formulated and asserted simultaneously. Intuitively, this ill-fitting is due to there not being a separation in the definitions between the protocol, which we want to be efficient, and the ideal DP mechanism, which we want to allow to be inefficient. For SIM$^+$-CDP (Definition 3.3) there is no such dissonance in the modeling of the protocol since it is already formulated using the ideal/real paradigm. Here the troubles lie rather in the details of the definition, which we will see are too restrictive to allow the notion to be fulfilled by emulating most common SDP mechanisms. This is fundamentally due to SIM$^+$-CDP requiring perfect correctness in the MPC protocol, which together with a demand for protocols running in strict polynomial time rules out any SDP mechanism that uses noise that is not samplable exactly in strict polynomial time. Whereas SIM$^+$-CDP could be achieved by using a finite version of standard SDP mechanisms, for instance using the mechanisms introduced in [2], it does mean a less direct realisation of GED, since the intuition is still to, say, 'use MPC to run the geometric mechanism'. Therefore, in Section 4, we propose an adapted version of SIM$^+$-CDP, calling it SIM*-CDP, which indeed can be satisfied by emulating standard CDP mechanisms due to a relaxation to computational correctness. Other large changes from SIM$^+$-CDP include using the UC (Universal Composability) security framework [10] instead of standalone security [9, 37] and allowing other ideal functionalities than secure function evaluation. [5] We prove that SIM*-CDP is of incomparable strength to SIM$^+$-CDP, meaning that there are in both ways computational tasks that can be solved with one but not the other, and that (like SIM$^+$-CDP) SIM*-CDP is strictly stronger than SIM-CDP and IND-CDP.

Implementing a protocol satisfying the new definition. To demonstrate the advantages of SIM*-CDP, we implement a generic protocol for satisfying SIM*-CDP for the ideal functionality computing the truncated geometric mechanism. In particular, we analyse the noise sampling protocol of [30], adjust it to use mixed circuits for improved efficiency and give a very direct proof that the resulting protocol satisfies SIM*-CDP. Further, we implement the protocol and thereby present the first implementation of the protocol of [30] and simultaneously the first implementation of the truncated geometric mechanism with active security. Finally, we show how to use the protocol for computing integer inner-products with CDP and accuracy equal to that of the central model and benchmark the implementation, showing its practical efficiency. This treatment might be of independent interest, perhaps primarily due to our considerations relating to that the function sensitivity of the inner-product is dependent on the input domain of the corrupted party, thus creating a need for input validation. We note that whilst the definitions of CDP remain relatively unchanged when going from passive to active corruptions, the concrete privacy proof of a given protocol often changes significantly (as does the practical efficiency of its implementation) thereby the simplicity of our analysis in this more complicated setting showcases the usability of SIM*-CDP.

Contributions:

- We identify aspects of existing CDP definitions that make them an unnatural fit to the approach of generic emulation of a central trusted dataholder that computes an inefficient SDP mechanism. Therefore, we present a new version of SIM$^+$-CDP, which we call SIM*-CDP, and formally relate it to previous definitions (Sections 3 and 4).
- We demonstrate the usability of the SIM*-CDP definition by showing how it can be satisfied with the truncated geometric mechanism by proving that the efficient MPC protocol by [30] for sampling geometric noise satisfies our definition (Sections 5 and 6).
- We improve the efficiency of the protocol by using mixed circuits and use the protocol to compute two-party inner-products with CDP and active security, to the best of our knowledge being the first to do so with accuracy equal to that in the central model. Our open-sourced implementation is the first implementation of the noise sampling protocol of [30]. We provide benchmarks of the implementation and thereby show that it is efficient in practice (Section 7).

Related works. The first work that aims to emulate a central trusted party for DP by use of MPC is *Our data, ourselves* [26], where a protocol is proposed for computing sums with security against active adversaries corrupting less than a third of the parties, a part of which is a method for distributed noise generation. Following [26], other works have also proposed noise sampling protocols for DP in an MPC setting [1, 14, 30, 71] and the work most related to ours is EIKN [30, 31]. EIKN gives an efficient MPC protocol for sampling an approximate truncated geometric distribution, which we use in this work. The mechanism is analysed with respect to IND-CDP, however the privacy proofs given are

[4] For more details on this, see Appendix A. The core observation there is that they cannot be computed exactly in strict polynomial time on a finite computer due to having probability distributions containing densities that are not an inverse polynomial power of 2.

[5] We underscore that the merit of our new definition is not that it allows studying new scenarios or is to be preferred over previous definitions in all cases, indeed there are many cryptographic tasks for which UC-secure protocols are missing or for which it is not the most desirable framework to use. Rather the merit is that for settings where UC-secure protocols are readily available, then we have a formulation that takes advantage of that to give results that are both stronger and easier to obtain.

only for honest majorities and thus do not apply to the two-party case [30, 31]. In a recent work [50], efficient noise sampling protocols for passive corruptions and dishonest majorities are provided. It is noted in passing that the protocols can easily be made secure against active adversaries by implementing them in a framework with active security but the type of CDP this would result in is not discussed. Our proposed SIM*-CDP definition offers an immediate answer to that. The work of [14] proposes a method for performing Bernoulli trials that is asymptotically superior to the one we use however their method relies on implementing oblivious data structures hence making it unsuitable for direct combination with the secret-sharing-based MPC schemes that we use. Further, avoiding reasoning about oblivious data structures greatly simplifies our proofs and the disposition of later sections, allowing us to focus more on details relating to the CDP definitions.

Another line of work that is of relevance to ours due to it dealing with combining definitions of security for MPC schemes and DP is the serie of papers considering MPC with differentially private leakage [42, 45, 59], where the idea is to improve the efficiency of an MPC protocol by allowing the protocol execution itself (not the result) to leak some extra information, but to restrict this leakage to be differentially private. Whilst the task solved in this line of work is quite different from the one we study, there are similarities in the formalities, which we discuss after having introduced SIM*-CDP.

2 Preliminaries and Notation

For a natural number n, let $[n] := \{1, \ldots, n\}$. Let $\mathbb{N}^{-1}$ denote $\{1/n : n \in \mathbb{N}\}$.

2.1 Secure Computation

We now briefly introduce necessary terminology regarding secure multiparty computation, for a slightly more thorough introduction, see Appendix B. A protocol is described as a set of interactive machines. For our purposes, it is not important exactly how those machines are formalised but for concreteness, we will think of interactive Turing Machines, which are non-uniform unless otherwise stated. We say that an algorithm (or machine or protocol) is *efficient* if it is PPT, meaning it is probabilistic and runs in strict polynomial time. We quantify the security by a protocol by a computational security parameter κ.[6] We consider both active and passive corruptions but assume they are static. For a protocol $\pi = \{P_1, ..., P_n\}$ and a set $C \subset [n]$, let $\{P_C\}$ denote $\{P_i : i \in C\}$ and let $\{P_{-C}\}$ denote $\{P_i : i \notin C\}$. The information available to the coalition C of parties in the protocol is formalised in their view, as defined below. The reason for exchanging $\{P_C\}$ with $\{\tilde{P}_C\}$ is to model active corruptions.

*Definition 2.1 (*VIEW*, reformulation from [4]).* Let $\pi = \{P_1, ..., P_n\}$ be a protocol and $\mathcal{A}$ be an adversary corrupting a set $C \subset [n]$ of parties. For fixed inputs $D = (D_1, ..., D_n) \in \mathcal{D}$, the view in π of the corrupted parties $\{\tilde{P}_C\}$, denoted $\mathsf{VIEW}^{\mathcal{A}}_{\pi,C}(D)$, is defined as the random variable containing the inputs of the parties in C, their random coins and the messages that they receive during the execution of the protocol $\{\tilde{P}_C\} \cup \{P_{-C}\}$ on inputs D. The randomness is over the random coins of the honest parties $\{P_{-C}\}$.

Often it is clear from context what parties the adversary corrupts (for instance in a symmetric two-party protocol) and then we omit C from notation. For defining secure computation of protocols we use the standard definitions in the ideal/real-paradigm, in both the standalone [9, 37, 56] and UC frameworks [10, 21]. Very shortly one can say that security is defined by formulating an ideal world in which an incorruptible trusted central party, an *ideal functionality*, performs all computations (and is secure by definition) and then a real-world protocol is deemed secure if no efficient distinguisher can distinguish it from the ideal world.

2.2 Differential Privacy

The notion of *differential privacy (DP)* [25, 28] considers a probabilistic algorithm, or *mechanism*, that maps *databases*, i.e. sets of elements from some data universe χ, to some output range R. We think of databases as ordered sets of some fixed (public) size N, and thus a database D is an element of $\mathcal{D} := \chi^N$. We say that two databases D, D' are *adjacent* if they differ in at most one element. There are however many other adjacency notions that may be more suitable to a given use case, perhaps especially when considering different types of distributed settings, so it is important to note that the definition of (S)DP in itself is agnostic to the choice of adjacency notion, just like all CDP definitions considered in this paper.[7]

Definition 2.2 (Adjacency notion). An adjacency notion ADJ on the dataset domain $\mathcal{D}$ is a set in $\mathcal{D} \times \mathcal{D}$ that is symmetric, i.e. if $(D, D') \in \mathsf{ADJ}$ then so is (D', D), and $\forall D \in \mathcal{D}, (D, D) \in \mathsf{ADJ}$. If $(D, D') \in \mathsf{ADJ}$ the we say that D and D' are adjacent with respect to ADJ.

We recall the standard definition of SDP (reformulation of [25]):

Definition 2.3 ((ε, δ)-SDP [25, 28]). A probabilistic algorithm $\mathcal{M} : \mathcal{D} \to R$ is (ε, δ)-differentially private (SDP) if for all pairs (D, D') of adjacent databases in $\mathcal{D}$ and all subsets S of R,

$$\mathbb{P}(\mathcal{M}(D) \in S) \leq e^{\varepsilon}\mathbb{P}(\mathcal{M}(D') \in S) + \delta, \tag{1}$$

where the probability is over $\mathcal{M}$'s internal coin tosses.

As is standard in cryptography, we typically consider not single mechanisms but rather *ensembles* of them and index the individual mechanisms within an ensemble by a security parameter $\kappa \in \mathbb{N}$. The security parameter is used to quantitatively relate properties of a mechanism or protocol (for instance the success probability of a given type of adversary) to specific parameter choices. This approach applies also to DP, and we therefore often allow DP parameters to depend on κ, letting $\varepsilon_\kappa = \varepsilon(\kappa), \delta_\kappa = \delta(\kappa)$ denote sets of parameters. Then we abuse notation by saying (for instance) that the ensemble $\mathcal{M} = \{\mathcal{M}_\kappa\}_{\kappa \in \mathbb{N}}$ is $(\varepsilon_\kappa, \delta_\kappa)$-SDP if for all large enough κ, $\mathcal{M}_\kappa$ is $(\varepsilon_\kappa, \delta_\kappa)$-SDP. Since the introduction of a dependence on κ

[6] Many of our results will be quantified by κ even if they also hold with respect to an equal *statistical* security parameter since statistical security implies computational security.

[7] In particular, since the focus of this paper is on the CDP definition with respect to GED and the choice of adjacency notion does not influence the merit of a CDP definition over another, it is for our discussions not relevant what adjacency notion one chooses to work with. On the practical side, there is however a potentially large difference in how well an adjacency notion fits together with a given MPC technique or setting, but these matters concern the realisation of GED rather than what the approach of GED means for the definition of CDP.

is most directly a consequence of relying on cryptographic guarantees in the mechanism design, it might rightfully be more closely associated with CDP than with SDP. We do however find that it is convenient to include this dependence also when discussing SDP, partly because it allows a more direct comparison to CDP and partly because the security parameter readily arise during the design of algorithms, also for SDP mechanisms.

The formulation of SDP above is often called *approximate SDP*, whereas it is called *pure SDP* if δ is fixed to 0. DP is typically studied in what is called the *central model*, of which an illustration can be found in Figure 1. In the central model, the database is simply a set of rows, each of which consists of information about one individual, called a *data subject*. These data subjects send their data to a trusted *dataholder* (without noise) that then computes a mechanism on the accumulated data and then releases the result to an untrusted *data analyst*. In this work, we rather consider DP in the *two-party model* [60, 64] where each data subject holds two database rows (x_i, y_i), each of which is sent to one of two computational parties (or *servers*) that then store their respective row into their database ($\mathbf{x}$ and $\mathbf{y}$ respectively) in the clear. Then these two servers together wish to compute the query f on the concatenation of their databases $D := \mathbf{x}||\mathbf{y}$, both learning the result, and they wish to do this in a differentially private manner with respect to their database. An illustration of this model can be seen in Figure 2.[8] For more details on models of DP in multiparty settings, see [63].

When discussing DP mechanisms, it is critical to consider the usefulness of the mechanism for approximating the query function f. We do this by using the following notion of usefulness, as defined via a utility function.

Definition 2.4 (Utility function [7, 35]). A utility function is an efficiently computable deterministic function $u : \mathcal{D} \times \mathcal{R} \to \{0,1\}^*$. A mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ is α-useful for u if for all $D \in \mathcal{D}$:

$$\mathbb{P}_{z \leftarrow \mathcal{M}(D)}(u(D,z) = 1) \geq \alpha. \tag{2}$$

A mechanism α-useful for u is said to solve the task (α, u).

A specific utility function we will consider is that which induces the notion of $(s,\alpha)-$*additive-usefulness* for a query function f, namely $u(D,z) = 1$ iff $|f(D) - z| \leq s$. Many popular DP mechanisms (such as the Gaussian, Laplace and geometric mechanisms) work computing the query function and then add noise of a specific distribution calibrated after the *sensitivity* of f (how much any single database entry can change the function evaluation). In this work, we consider this change only in the sense of l_1-distance.

Definition 2.5 (l_1-sensitivity). Let $f : \mathcal{D} \to \mathcal{R}$ be a deterministic function, where $\mathcal{R}$ is a vector space on which the l_1-norm $||\mathbf{v}||_1 := \sum_i |v_i|$ is defined, and ADJ be an adjacency notion on $\mathcal{D}$.

[8] We note that the two-party model is slightly but significantly different from the *two-server/multi-server models* [5, 17], primarily in that those models do not allow any server to have any part of the input dataset in the clear. This difference is of practical relevance because it means the models are suitable for different scenarios. The two-party model is mostly meant for *joint computation* between two entities each holding their own dataset (which may have been collected over time and without respect to the function evaluation in question) whereas the two-server model is rather tailored towards *data collection*, where one or more entities are collecting the data specifically for the purpose of performing the computation but wish to do so in a way that they never see any part of the dataset in the clear.

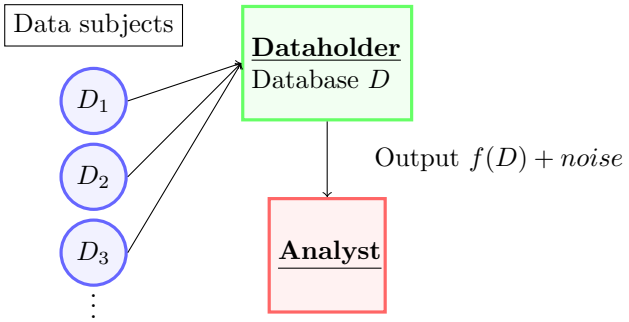

Figure 1: In the *central model*, the data subjects trust the data holder with their data (D_i) but wish to keep it secret from an (possibly adversarial) analyst learning the (possibly noisy) function evaluation.

The l_1-sensitivity of f with respect to ADJ, denoted Δf, is defined as

$$\Delta f := \max_{(D,D') \in \text{ADJ}} ||f(D) - f(D')||_1. \tag{3}$$

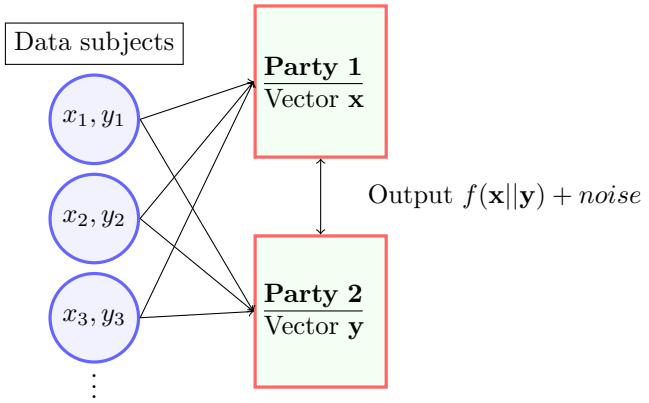

Figure 2: In the *two-party model*, the data subjects trust two different data holders, which we call *parties*, with a different part of their data, but not with the part of the data that they send to the other data holder. In the end both parties learn the noisy function evaluation. Thus, in a sense, each party plays both the role of a data holder and a data analyst.

2.3 Mixed Binary-arithmetic MPC Schemes

In our definitions, we rely on general-purpose MPC schemes with active security. In particular, we work with MPC protocols with restricted computation domain, either in $\mathbb{F}_p$ for arithmetic or $\mathbb{F}_{2^k}$ for binary circuits. For a discussion of active security in these schemes, we refer to Appendix E. In general, MPC schemes in $\mathbb{F}_p$ provide fast algorithms for addition and multiplication. In contrast, in $\mathbb{F}_{2^k}$, comparisons, bit-wise operations, and non-linear functions can be evaluated cheaply. However, storing larger integers results in substantial overhead, and evaluating arithmetic circuits in the binary domain incurs costs depending on the encoded values' bit size.

Several works have proposed solutions to convert shares between computation domains. First, in ABY [24], the authors propose a semi-honest two-party MPC scheme that allows switching between the binary, arithmetic, and garbled circuit domains (Garbled Circuits allow computation of binary circuits with low communication rounds). More recently, Rotaru and Wood introduced *doubly-authenticated bits* [66] and an efficient procedure to securely sample secret bits in the arithmetic and binary domain in malicious settings. Given the shares of an unknown random bit $([\![b]\!]_2, [\![b]\!]_p)$ we can transfer shared bits from the binary to the arithmetic domain by computing the mask $m \leftarrow \mathsf{Reconstruct}([\![x]\!]_2 \oplus [\![b]\!]_2)$ and setting $[\![x]\!]_p \leftarrow m + [\![b]\!]_p - 2 \cdot m[\![b]\!]_p$. Similarly, converting from arithmetic to binary masks the value by addition and evaluates subtraction in the binary domain. The conversion from the arithmetic to the binary domain gets more expensive, depending on the field size. Subsequent work introduced *extended doubly-authenticated bits (eda-bits)* [33], where masking values are shared along with their binary decomposition in the respective domains. The eda-bits represent an improvement in efficiency when converting larger values, and [33] presents dedicated protocols to speed up comparisons in $\mathbb{F}_p$.

3 CDP in the Two-party Model

We now briefly overview the literature on CDP in the two-party model and argue why it is desired to look for new definitions. For more details on existing definitions and how they relate, see [63].

3.1 Existing CDP Definitions for Protocols

The formal study of both SDP and CDP in the two-party and multi-party models is initiated in [4, 64], where three definitions of two-party CDP are proposed. These are formulated for the two-party case but the definitions trivially extend to the multi-party case. We also follow this convention. The notion of SDP in the central model is extended to interactive protocols by requiring that the view of the adversary is an SDP mechanism with respect to the input of the honest party. In [4] it is established that there are computational tasks for which the maximum utility in the two-party SDP model is strictly lower than in the central model and therefore there is a need to relax SDP to CDP. The CDP definitions come in two distinct variations, based on how they formalise a protocol execution 'looking SDP' to a computationally bounded party. The first variation is called *indistinguishability-based* and changes the demand that the output distributions of the mechanism are close on adjacent inputs to that this must only hold for all PPT distinguishers acting on the mechanism output. The second variation of CDP is called *simulation-based* and here a mechanism is deemed CDP if there exist an SDP mechanism from which it is computationally indistinguishable. Below we include a reformulation of the definition of indistinguishability-based CDP.[9]

Definition 3.1 (IND-CDP for protocols, reformulation from [4, 68]). We say that a 2-party protocol π is $(\varepsilon_\kappa, \delta_\kappa)$-IND-CDP if for all efficient adversaries $\mathcal{A}$ corrupting at most one party, for all efficient distinguishers T, every sufficiently large κ and for all D, D' adjacent with respect to the inputs of the honest party, we have

$$\mathbb{P}\left(T\left(\mathsf{VIEW}_\pi^{\mathcal{A}}(D)\right) = 1\right) \leq e^{\varepsilon_\kappa}\mathbb{P}\left(T\left(\mathsf{VIEW}_\pi^{\mathcal{A}}(D')\right) = 1\right) + \delta_\kappa. \quad (4)$$

The probabilities are taken over the randomness in π, $\mathcal{A}$ and T.

It is noted in MPRV that if $\delta_\kappa = 0$ then IND-CDP is equivalent to pure SDP. IND-CDP was originally formulated with δ_κ fixed as negligible but the version with non-negligible δ_κ has also seen practical use, for instance, EIKN [30]. In the two-party model there are two different main formulations of simulation-based CDP, which we include below. These were introduced originally with $\delta_\kappa = 0$ but similarly have been used with larger δ_κ also [5].

Definition 3.2 (SIM-CDP for protocols, reformulation from MPRV). We say that a 2-party protocol π is $(\varepsilon_\kappa, \delta_\kappa)$-SIM-CDP if for all efficient adversaries $\mathcal{A}$ corrupting at most one party, for all efficient distinguishers T and D, D' adjacent with respect to the inputs of the honest party, there exists an ensemble $\{\mathcal{M}_\kappa(\cdot)\}_{\kappa\in\mathbb{N}}$ of $(\varepsilon_\kappa, \delta_\kappa)$-SDP mechanisms $\mathcal{M}_\kappa : \mathcal{D} \to \mathcal{R}_\kappa$ such that for every sufficiently large κ and every $D \in \mathcal{D}$ of size polynomial in κ, it holds that $\mathsf{VIEW}_\pi^{\mathcal{A}}(D)$ and $\mathcal{M}_\kappa(D)$ are indistinguishable to T.

Definition 3.3 (SIM$^+$-CDP, Reformulation of MPRV). Let u be a utility function. A 2-party protocol π is $(\alpha, \varepsilon_\kappa, \delta_\kappa)$-SIM$^+$-CDP for u if there exists an $(\varepsilon_\kappa, \delta_\kappa)$-SDP mechanism $\mathcal{M}$ such that:

- the mechanism $\mathcal{M}$ is α-useful for u;
- π is a secure protocol for the functionality $\mathcal{M}$ as per Definition B.2 (standalone security with perfect correctness, efficient protocols and a potentially inefficient simulator).

3.2 Relations Between CDP Definitions

There is substantial literature on how the CDP definitions relate to each other and although the relations are far from tightly characterised, the rough picture is quite clear. For the parameter regimes for which the definitions were originally proposed ($\delta_\kappa = \mathsf{negl}(\kappa)$ in IND-CDP and $\delta_\kappa = 0$ in the others), it was shown in MPRV that any protocol that is $(\varepsilon_\kappa, 0)$-SIM$^+$-CDP is also $(\varepsilon_\kappa, 0)$-SIM-CDP and similarly $(\varepsilon_\kappa, 0)$-SIM-CDP implies $(\varepsilon_\kappa, \mathsf{negl}(\kappa))$-IND-CDP. On the other hand, there are tasks that can be solved with $(\varepsilon_\kappa, 0)$-SIM-CDP that cannot be solved with $(\varepsilon_\kappa, 0)$-SIM$^+$-CDP. It was long unknown if there is a similar separation between $(\varepsilon_\kappa, \mathsf{negl}(\kappa))$-IND-CDP and $(\varepsilon_\kappa, 0)$-SIM-CDP but in 2023 such a task was found [35]. The definitions have mostly been related to each other by either considering a fixed task and showing that there are no complexity assumptions under which that task can lead to a separation or by considering a fixed complexity assumption and showing that there are no tasks that lead to a separation under that assumption alone [7, 41, 61]. There is also a line of work about finding minimal complexity assumptions under which (various types of) CDP can be separated from SDP via a specific task, like computing boolean functions or integer inner-products with a given accuracy [39, 40, 43, 44, 53]. Whereas the relationship between the involved DP definitions is quite well understood in these cases, one should note that the same does not hold generally for other classes of tasks or for more relaxed parameter regimes.

[9] Its original formulations [4, 64] differ slightly from one another, for instance in that [4] allows only passive corruptions and that MPRV lets the distinguisher be non-uniform. We consider these differences however to be of the sort making the definitions more two different instantiations of the same definition rather than two different ones. Note also that they both fix δ_κ as negligible (but non-zero) in κ.

3.3 Using Existing Definitions for GED

Each of the CDP notions above can be satisfied by a protocol that uses general-purpose MPC techniques to realise a functionality that computes an SDP mechanism, i.e. a protocol for GED. This has been shown for IND-CDP in [4, 68] and for SIM-CDP in MPRV. For SIM$^+$-CDP it is immediate, since there are general-purpose MPC schemes for the notion of secure computation used in SIM$^+$-CDP. We argue that GED results in guarantees that are fundamentally stronger than those in IND-CDP and SIM-CDP, therefore warranting a definition that captures them more closely, and that there are details in the SIM$^+$-CDP definition that make it inconvenient to work with for GED, although intuitively it is very suitable.

On using IND-CDP or SIM-CDP. One strength of GED for constructing CDP protocols is that one has guarantees about the behaviour of the protocol which exceed that of the adversarial view appearing DP. More precisely, the use of MPC allows guaranteeing *security* (dictating the influence an adversary may have on the protocol) and *correctness* (specifying the accuracy requirement of an honest execution). These properties are proven in the ideal/real paradigm (see Appendix B), that is, by specifying an ideal functionality that defines all of the desired properties of the protocol and then proving that the real protocol behaves almost the same. Here there arises a dissonance in intuition to the perspective in IND-CDP, since that notion considers only the real-world protocol. Therefore, when one uses IND-CDP together with GED, one has CDP as a property of the protocol in the real world, rather than in the ideal world with all other desired properties. This type of dissonance is smaller when it comes to SIM-CDP since the mechanism $\mathcal{M}$ in SIM-CDP is also in a way a description of the simulator and ideal functionality. The dissonance here, however, is that the formalisation of simulation is vastly relaxed in SIM-CDP, for instance in that the simulator has access to all private inputs.

On using SIM$^+$-CDP. SIM$^+$-CDP does not suffer the modeling-wise dissonance described above and neither does it poorly capture the guarantees granted by secure computation. The problem with using SIM$^+$-CDP in the context of GED lies rather in that some of the details in the definition are too restrictive, meaning that they rule our realising GED for many of the most fundamental SDP mechanisms. In particular, the SIM$^+$-CDP definition requires the real-world protocol π to run in strict polynomial time and simultaneously have perfect correctness, meaning that its output distribution in an honest execution is *identical* to that of the SDP mechanism in the ideal functionality. This implies that any protocol satisfying SIM$^+$-CDP must do so with respect to an SDP mechanism that can be computed *exactly* in *strict* polynomial time. Unfortunately, this rules out several commonly used SDP mechanisms, such as the Laplace or Gaussian mechanisms, which we now showcase with the example of the Laplace mechanism.

Impossibility of GED with the Laplace mechanism in SIM$^+$-CDP. The main question to ponder is whether there exists an efficient protocol that can realise the Laplace mechanism in SIM$^+$-CDP. Unfortunately, there is not.[10] To begin with, the support of the Laplace mechanism is the reals, meaning the output cannot even be written in strictly finite time. Thus we can note that any mechanism in the SIM$^+$-CDP definition must have a finite support. Further, even the (arguably) most Laplace-like such distribution, the geometric distribution [36] truncated to the output domain, cannot be realised in SIM$^+$-CDP in general, since it requires sampling probabilities that are not multiples of $2^{-poly(\kappa)}$ (for details on the impossibility of sampling certain distributions in strict polynomial time, see Appendix A). This means that in order to realise GED with distributions that cannot be sampled exactly in strict polynomial time (as is the case for the Laplace, geometric, Gaussian, discrete Gaussian distributions and truncated versions of them), there needs to be some slack introduced. This could be either in the shape of allowing a small distance between the output of the ideal functionality and that of the protocol (relaxing correctness) or relaxing the demand for strict polynomial time to expected polynomial time, as is argued in [13].

As remarked shortly in the introduction, one approach in practice could be to use SIM$^+$-CDP with non-zero δ_κ and have the ideal functionality compute an efficiently computable approximation of a standard SDP mechanism. Then if the SDP mechanism is $(\varepsilon_\kappa, \delta_\kappa)$-SDP and the approximation of it is at a statistical distance of δ'_κ, then it is easy to see that the protocol which securely realises this functionality (in the way required by SIM$^+$-CDP) is $(\varepsilon_\kappa, \delta_\kappa + \delta'_\kappa)$-SIM$^+$-CDP. That is, the approximation error can be added to the δ_κ. In many settings, this is a likely a suitable approach. One main drawback of it, however, is that the approximation error is fundamentally different from the δ_κ term in both cause and interpretation. Another drawback is the need to introduce these efficient approximations of standard mechanisms explicitly, rather than to handle all matters of approximation within the simulation argument.

4 A New Version of Simulation-based CDP in the Ideal/real Paradigm

4.1 Our New Definition, SIM*-CDP.

We now propose a new version of SIM$^+$-CDP, which we call SIM*-CDP and then discuss its relationship to previous definitions further.

Definition 4.1 ($(\varepsilon_\kappa, \delta_\kappa)$−SIM*-CDP). The two-party protocol π is $(\varepsilon_\kappa, \delta_\kappa)$-SIM*-CDP for the ideal functionality $\mathcal{F}$ and adjacency ADJ notion if π UC-realises $\mathcal{F}$ and for all ideal-world adversaries $\mathcal{S}$, the view of $\mathcal{S}$ is $(\varepsilon_\kappa, \delta_\kappa)$-SDP with respect to ADJ.

The main differences between SIM*-CDP and SIM$^+$-CDP are:

- *UC-security is used as security notion.*
- *The ideal functionality is variable (and can be reactive).*
- *Correctness is computational rather than perfect.*
- *The ideal-world adversary (simulator) must be efficient (strict PPT).*
- *The requirement of usefulness is removed from the CDP definition.*

We now expand on the motivation behind these changes.

Using UC-security. Although the standalone security framework is heavily used, in the last two decades the security analyses of many popular schemes have taken place in the more expressive

[10] We note that this invalidates the claims in [1] of achieving SIM$^+$-CDP for the (continuous, untruncated) Laplace mechanism. The protocol there does however seem to satisfy a relaxation of SIM$^+$-CDP, in line with the contents of Section 4, although that remains to be formally shown.

UC framework [10]. The main merit of this framework is that the security can be proven to be preserved under arbitrary composition of protocols, leading to a stronger notion of security and an increased modularity in security proofs. Thus, using UC security in the CDP notion is natural for cases where this (stronger) type of security is already achieved by the MPC scheme one intends to use. Further, as we will see below, this change of security framework also directly leads to many other benefits.

A variable ideal functionality. The ideal functionality used in SIM$^+$-CDP is fixed to be that of *secure function evaluation (SFE)*, i.e. the parties jointly compute an SDP mechanism (with abort). With regards to capturing what it means for a protocol to be CDP generally, this is a significant restriction as compared to IND-CDP and SIM-CDP, where CDP is defined without dependence on the functionality of the protocol. In particular, both IND-CDP and SIM-CDP allow direct modeling of reactive functionalities, and as such our new definition arguably lies closer to those definitions conceptually than SIM$^+$-CDP does, in that it is applicable to more functionalities than those that can be expressed as SFE [42, 46, 47]. On the practical side, one relevant reactive functionality is that of SFE with differentially private leakage as in, for instance, [42]. More details about the setting of SFE with DP leakage are found in Appendix C.

Computational correctness. Another positive consequence of changing security framework is that the correctness of the protocol is now (as is standard in UC-security) computational rather than perfect. As explained in the previous section, this relaxation allows the ideal functionality to sample inefficiently samplable distributions and still have an efficient protocol that realises it.

Efficient simulators. As one main goal of our new definition is to have it align closely to common practice in MPC, we choose to require efficient simulation. Whereas this does make fulfilling the definition harder, it also makes the definition stronger.

Not including usefulness in the definition. A final difference between SIM*-CDP and SIM$^+$-CDP is that we choose not to include the requirement for usefulness in the definition of CDP itself. This is done primarily to more closely correspond to how the matter of usefulness is handled for IND-CDP and SIM-CDP in MPRV, namely that the CDP definition is agnostic to the notion of usefulness (Definition 6 in MPRV [64]) and that usefulness is then added later (Definition 7 in MPRV). Another advantage of not having the usefulness as a part of the CDP definition is that one can choose to consider the usefulness simply of the ideal functionality (as is done in SIM$^+$-CDP) or to consider the usefulness of the protocol directly (as with IND-CDP and SIM-CDP in Definition 7 of MPRV) and then take, for instance, failure probabilities of the protocol into account.[11] The utility difference between the real protocol and the ideal functionality is however bounded to be negligible by the simulation argument, since the utility function is efficiently computable and if the utility difference was non-negligible then the utility function would serve as an efficient distinguisher between the real and ideal worlds.[12]

[11] For SIM$^+$-CDP one should note that the usefulness of the protocol is always the same as that of the ideal functionality unless there are active corruptions, due to the requirement of perfect correctness.

[12] A similar remark is made in [35].

Functionality $\mathcal{F}_{SFE}^{f}$

Parameters:

- A function $f = (f_1, f_2) : (\{0,1\}^*)^2 \to (\{0,1\}^*)^2$.

No corruptions:

- Upon $\mathbf{x}_1$ from P_1 and $\mathbf{x}_2$ from P_2, deliver $f_1(\mathbf{x}_1, \mathbf{x}_2)$ to P_1 and $f_2(\mathbf{x}_1, \mathbf{x}_2)$ to P_2.

Party P_c corrupted (P_h is honest):

- Upon $(\text{Input}, \mathbf{x}_h)$ from P_h and $(\text{Input}, \mathbf{x}_c)$ from P_c, send $f_c(\mathbf{x}_1, \mathbf{x}_2)$ to P_c.
- Upon $(\text{Deliver}, b)$ from P_c, if $b = 1$ then send $f_h(\mathbf{x}_1, \mathbf{x}_2)$ to P_h, otherwise send $\perp$.

Figure 3: The ideal functionality for SFE with abort.

To round this subsection off, we re-iterate the standard ideal functionality for SFE with abort, see Figure 3. In Section 6 we propose a protocol for realising this ideal functionality with the geometric mechanism as the functions f_1 and f_2 and prove it is SIM*-CDP in the presence of active corruptions.

4.2 Relating SIM*-CDP to Other Definitions

We now relate our new definition to existing ones. For all of the propositions, the proofs are delegated to Appendix D. We prove separations only when $\delta_\kappa = 0$ (or $\delta_\kappa = \text{negl}(\kappa)$, depending on the CDP notion), as is common in the literature, and leave extending the separations to other settings for future work.

Relation to SIM$^+$-CDP . There is no general hierarchy between SIM$^+$-CDP and SIM*-CDP, in the sense that there are both tasks that can be solved with SIM$^+$-CDP but not SIM*-CDP and the other way around. In one direction this is due to SIM*-CDP being more restrictive in that it demands UC-security instead of standalone security since there are well known results of functionalities that can be realised with standalone security but not UC-security unless certain setup assumptions are made [11]. In the other direction, SIM*-CDP is more relaxed than SIM$^+$-CDP with regard to the correctness of the protocol. In more formal terms, see the propositions below.

Proposition 4.2. *Using the plain UC model, i.e. without setup assumptions, and assuming that enhanced trapdoor permutations (see [37]) exist, there exists ε_κ for which there exists a task that is solvable with $(\varepsilon_\kappa, 0)$-SIM$^+$-CDP but not with $(\varepsilon_\kappa, 0)$-SIM*-CDP. This holds regardless of whether the utility requirement is placed on the real or the ideal protocol with respect to SIM*-CDP.*

Proposition 4.3. *Using the UC model with the setup assumption of a common reference string (CRS) (see, for instance, [11]) and with the utility in SIM*-CDP being considered in the ideal world (i.e. with regards to the utility of $\mathcal{F}$), there exists ε_κ for which there exists a task that is solvable with $(\varepsilon_\kappa, 0)$-SIM*-CDP but not with $(\varepsilon_\kappa, 0)$-SIM$^+$-CDP.*

Relation to SIM-CDP and IND-CDP. Just as with SIM$^+$-CDP, on the one side if a protocol is SIM*-CDP then it is SIM-CDP (and thus also IND-CDP, see [64]) but on the other side there are tasks that can be solved with SIM-CDP but not in SIM*-CDP. The second separation is a direct corollary of Proposition 4.2 due to that all

SIM$^+$-CDP protocols also are SIM-CDP protocols with unchanged parameters.

Proposition 4.4. *For any parameters $\varepsilon_\kappa, \delta_\kappa$, if a two-party protocol π is $(\varepsilon_\kappa, \delta_\kappa)$-SIM*-CDP, then it is also $(\varepsilon_\kappa, \delta_\kappa)$-SIM-CDP.*

Corollary 4.5 (of Proposition 4.2). *Using the plain UC model, i.e. without setup assumptions, and assuming that enhanced trapdoor permutations (see [37]) exist. Then there exists ε_κ for which there exists a task that is solvable with $(\varepsilon_\kappa, 0)$-SIM-CDP but not with $(\varepsilon_\kappa, 0)$-SIM*-CDP. This holds regardless of whether the utility requirement is placed on the real or the ideal protocol with respect to SIM*-CDP.*

Relation to MPC-with-DP-leakage definition in [42]. Within the literature on relaxing definitions of secure computation by allowing there to be non-negligible information leakage during protocol execution as long as this leakage is SDP (for a longer discussion on such protocols, see Appendix C), there is a definition (Definition C.1) that, like SIM*-CDP, uses UC-security and defines a CDP property of a protocol. The fundamental difference between SIM*-CDP and that definition is that their definition is fixed for a given ideal functionality and only a specific part of the view of the ideal-world adversary $\mathcal{S}$ is required to be SDP, whereas SIM*-CDP is defined for arbitrary ideal functionalities and the entire view of $\mathcal{S}$ must be SDP. Thus SIM*-CDP can be seen as both a restriction of the definition in [42] (where one requires the remaining parts of $\mathcal{S}$'s view to be SDP also) and as a generalisation of it since the ideal functionality is left variable. From another point of view, the definitions try to solve two distinct problems, but one can suitably consider the need we see to propose an alternative CDP definition to those of (say) IND-CDP and SIM$^+$-CDP as being the analog in general CDP to the motivation in [42] for giving a definition separate to those of [45, 59].

4.3 A More General Definition, SIM$^\circ$-CDP

The core idea of SIM$^+$-CDP and SIM*-CDP is the same (requiring the protocol to realise an SDP ideal functionality) and this opens up a wide space of such definitions since there is an abundance of different notions of secure computation in the MPC literature. One can for instance vary correctness, robustness or efficiency requirements for the different involved entities. This suggests a generalised definition of which SIM$^+$-CDP, SIM*-CDP and other natural variations are instantiations of. Below we formulate such a generalised definition and call it SIM$^\circ$-CDP.

Definition 4.6 (SIM$^\circ$-CDP). We say that a two-party protocol π is $(\varepsilon_\kappa, \delta_\kappa)$-SIM$^\circ$-CDP with respect to ideal/real security notion SEC for the ideal functionality $\mathcal{F}$ and adjacency notion ADJ if π realises $\mathcal{F}$ in the sense of SEC and for all ideal-world adversaries $\mathcal{S}$, the view of $\mathcal{S}$ is $(\varepsilon_\kappa, \delta_\kappa)$-SDP with respect to ADJ.

In light of this definition, the bulk of the discussion in this section can be seen as concerning the ways in which we regard the specific choice of security notion in SIM$^+$-CDP as being inconvenient with respect to GED. We are aware of only one other used instantiation of SIM$^\circ$-CDP and that is in [5] where SIM$^\circ$-CDP is instantiated using standard standalone security but with computational correctness. That CDP notion is stronger than SIM$^+$-CDP in that it requires efficient simulators but weaker in the sense of having relaxed correctness. The protocol presented in [5] is not SIM$^+$-CDP (imperfect correctness is needed) and neither is it SIM*-CDP (since no UC security proof is given).

5 A SIM*-CDP Version of the Geometric Mechanism

To demonstrate the use of our new definition, we now go through in detail how to satisfy it for the standard SFE ideal functionality with the truncated geometric mechanism as the function. Conceptually, this is very simple; one can simply use any PPT algorithm that samples a distribution with a sufficiently small statistical distance to a truncated geometric distribution and then compute that algorithm in MPC via some general-purpose, active secure, protocol. It is however worth considering hurdles that arise in the details, such as how to handle the mechanism's dependence on the query function, having a query function whose sensitivity depends on the inputs of both parties and the consequences of working over a finite field. One core step is, naturally, to sample a distribution that is close to a truncated geometric distribution. Sampling algorithms for such distributions can be found in [2, 30, 36], however, the truncation is to a range between 0 and some fixed positive integer. The results and methods however extend to $\mathbb{Z}_q$, and general queries of bounded magnitude.

Definition 5.1 (Truncated geometric distribution). Define the truncated geometric distribution $Z \sim Geo_{q,\lambda}(\tilde{f})$ centered at $\tilde{f} \in \mathbb{Z}_q$, truncated to $\mathbb{Z}_q := [\lceil -q/2 \rceil, \lfloor q/2 \rfloor)$, by its probability mass function:

$$p_Z(z) = \frac{e^{1/\lambda} - 1}{e^{1/\lambda} + 1} e^{\frac{-|z-\tilde{f}|}{\lambda}} \tag{5}$$

for $z \notin \{\lceil -q/2 \rceil, \lceil q/2 - 1 \rceil\}$, and

$$p_Z(z) = \frac{1}{e^{1/\lambda} + 1} e^{\frac{-|z-\tilde{f}|}{\lambda}} \tag{6}$$

for $z \in \{\lceil -q/2 \rceil, \lceil q/2 - 1 \rceil\}$.

Definition 5.2 (Range-truncated geometric mechanism). Let $\lambda \in \mathbb{N}^{-1}$ and let $f : \mathcal{D} \to \mathbb{Z}_q$ be a deterministic function. The Range-truncated geometric mechanism (RTGeo) over $\mathbb{Z}_q$ for f is defined as

$$\mathcal{M}^{q,f,\lambda}_{RTGeo}(D) := Geo_{q,\lambda}(f(D)). \tag{7}$$

It is easy to verify that $\mathcal{M}^{q,f,\lambda}_{RTGeo}(D)$ is an $(\varepsilon, 0)$-SDP mechanism as long as $\lambda = \frac{\varepsilon}{\Delta f}$. In line with [2], we only allow $\lambda \in \mathbb{N}^{-1}$, in order to avoid the need to represent real numbers, and this also implies $\varepsilon \in \mathbb{N}^{-1}$. Whereas $\mathcal{M}_{RTGeo}$ gives SDP, it is inconvenient to sample the noise distribution directly, partly because it requires knowledge of $f(D)$ and partly because it may not be efficiently samplable. Therefore we consider the following mechanism.

Definition 5.3 (Subrange-truncated geometric mech.). Let $B \in \{1, \ldots, \lceil q/2 \rceil - 1\}$ and $\lambda \in \mathbb{N}^{-1}$. Let the Subrange-truncated geometric (SRTGeo) mechanism over $\mathbb{Z}_q$ with noise truncation to $\mathbb{Z}_{2B}$, for a function $f : \mathcal{D} \to \mathbb{Z}_q$, be defined as $\mathcal{M}^{2B,f,\lambda}_{SRTGeo}(D) := f(D) + Geo_{2B,\lambda}(0)$, with the addition performed over $\mathbb{Z}_q$.

In the simple lemma below we give a bound on the statistical distance between the two mechanisms we have introduced this far. The proof, as the proofs of all other lemmas, is found in Appendix D. We note that we need to introduce a bound on the absolute value of

the query function, so as to not have the sensitivity of the function be affected by the modular arithmetics.

Lemma 5.4. *Let* $f^{max} := \max_{D\in\mathcal{D}}|f(D)|$, $B \in \mathbb{N}$, $\lambda \in \mathbb{N}^{-1}$ *and* $q > 2f^{max} + 2B$. *Then the statistical distance between* $\mathcal{M}^{2B,f,\lambda}_{SRTGeo}(D)$ *and* $\mathcal{M}^{q,f,\lambda}_{RTGeo}(D)$ *for all* $D \in \mathcal{D}$ *is at most* $e^{-B/\lambda}$.

We are now one step closer to a functionality that can be efficiently realised, since the noise sampling is no longer dependent on the function evaluation and the support of the noise is potentially much smaller than the entire $\mathbb{Z}_q$ and the support of f. The trouble still remains that the probabilities might not be negative polynomial powers of two. In [26, 30] it is presented distributions that can be exactly sampled under this constraint and that have a small statistical distance from a truncated geometric distribution. We use the procedure FDL (*Finite-range Discrete Laplacian*) introduced in EIKN [30].

*Definition 5.5 (*FDL *function and procedure).* Let $\mathbf{r} \in \{0,1\}^{Bd+1}$ be independent fair coins and $0 < e^{-1/\lambda} < 1$. Let $\hat{\alpha}^1 \leftarrow \frac{1-e^{-1/\lambda}}{1+e^{-1/\lambda}}$ and $\hat{\alpha}^i \leftarrow 1-\hat{\alpha}^1$ for $i = 2, ..., B$ be public parameters. Let $\oplus$ and $\wedge$ denote addition and multiplication over the binary field and let $\vee$ be shorthand for computing the OR operation by using binary addition and multiplication. Let all other operands be defined as normally over $\mathbb{Z}_q$. Define the function $\mathsf{FDL}_{\lambda,B,d} : \{0,1\}^{Bd+1} \to \mathbb{Z}_{2B} \subseteq \mathbb{Z}_q$ by the procedure in Algorithm 1. Let $\alpha = (\alpha_1, \alpha_2, ...)$ be the bit decomposition of $\hat{\alpha}$. The subprocedure $\mathsf{Ber}_{\hat{\alpha}} : \{0,1\}^d \times \{0,1\}^d \to \{0,1\}$ for generating approximate Bernoulli trials with parameter $\hat{\alpha}$ using a randomness seed in $\{0,1\}^d$ is defined by the procedure in Algorithm 2.

Procedure FDL
Input: $\mathbf{r} \in \{0,1\}^{Bd+1}$

(1) Sample B approximate Bernoulli trials $\beta_i \leftarrow \mathsf{Ber}_{\hat{\alpha}^i}((r_{d(j-1)+1}, ..., r_{dj}))$ for $i = 1, ..., B$.
(2) For $i = 1, ..., B$: set $c_i \leftarrow \vee_{j=1}^{i} \beta_j$.
(3) Set $l \leftarrow B - \sum_{i=1}^{B} c_i$.
(4) Set $\sigma \leftarrow 2 \cdot r_{Bd+1} - 1$.
(5) Output $\sigma \cdot l$.

Algorithm 1: The algorithm description for the FDL procedure.

Procedure Ber
Input: $\mathbf{r} \in \{0,1\}^d$, $\alpha \in \{0,1\}^d$

(1) For $i = 1, ..., d$, set $c_i \leftarrow \alpha_i \oplus r_i$.
(2) For $i = 1, ..., d$, set $e_i \leftarrow \vee_{j=1}^{i} c_j$.
(3) For $i = 1, ..., d$, set $v_i \leftarrow e_i \oplus e_{i-1}$, with $e_0 \leftarrow 0$.
(4) Set $\beta \leftarrow 1 \oplus_{i=1}^{d} (r_i \wedge v_i)$ and output β.

Algorithm 2: The algorithm description for the Ber procedure.

Note that FDL is an exact method for turning $Bd + 1$ fair coins into a sample of a distribution that is statistically close to a truncated geometric one. It is clear that if the number of fair coins is polynomial in κ then FDL runs in strict polynomial time. With some abuse of notation, we use FDL to denote both the procedure and the probability distribution it generates upon being given fair coins.[13]

*Definition 5.6 (*FDL *mechanism).* Let $B \in \{1, \ldots, \lceil q/2 \rceil - 1\}$. Let the Finite-range Discrete Laplace (FDL) mechanism over $\mathbb{Z}_q$ for a function $f : \mathcal{D} \to \mathbb{Z}_q$ be defined as $\mathcal{M}^{\lambda,B,d,f}_{\mathsf{FDL}}(D) := f(D) + \mathsf{FDL}_{\lambda,B,d}$, with the addition performed over $\mathbb{Z}_q$.

The following lemma is proven in EIKN [30].

Lemma 5.7. *Let* $f^{max} := \max_{D\in\mathcal{D}}|f(D)|$, $q > 2f^{max} + 2B$ *and* $B \in \{1, \ldots, \lceil q/2 \rceil - 1\}$. *If* FDL *is given independent fair coins and all the arithmetics are done over* $\mathbb{Z}_q$, *then the statistical distance between* $\mathcal{M}^{\lambda,B,d,f}_{\mathsf{FDL}}(D)$ *and* $\mathcal{M}^{2B,f,\lambda}_{SRTGeo}(D)$ *is at most* $B \cdot 2^{-d}$.

Further, we have that $\mathcal{M}^{q,f,\varepsilon/\Delta f}_{RTGeo}(D)$ is a useful approximation of f, as we show in the following lemma.

Lemma 5.8. *Let* $q > 2f^{max} + 2B$, $B \in \{1, \ldots, \lceil q/2 \rceil - 1\}$. *Let* $f : \mathcal{D} \to \mathbb{Z}_q$ *be an arbitrary deterministic function with* $f^{max} := \max_{D\in\mathcal{D}}|f(D)|$ *and let* $\hat{f}(D) := \mathcal{M}^{q,f,\lambda}_{RTGeo}(D) : \mathcal{D} \to \mathbb{Z}_q$. *Then* $\hat{f}$ *is* $\left(\nu, \frac{2e^{-1/\lambda}}{e^{-1/\lambda}+1} e^{-\nu/\lambda}\right)$*-additive-useful for* f *for any positive integer* ν.

6 A Protocol for the FDL Mechanism

From the previous section, we know that the FDL mechanism is statistically close to the Range-truncated geometric mechanism ($\mathcal{M}_{RTGeo}$), which is pure SDP, and that this holds under some restrictions on the query function and on the parameter choices. At the same time, it is immediate that $\mathcal{M}_{RTGeo}$ is statistically close to the untruncated geometric mechanism (i.e. when the noise is not truncated and that the modular arithmetics thus might cause overflows), as long as the value of the query function is somewhat far away from $\pm q/2$. Therefore, there is a choice to be made regarding which mechanism one chooses to have in the ideal functionality (call this the *ideal mechanism*), given that we will have the protocol compute the FDL mechanism via general-purpose MPC. The trade-off in this choice is that having $\mathcal{M}_{RTGeo}$ as the ideal mechanism will lead to $(\varepsilon_\kappa, 0)$-SIM*-CDP as long as the statistical distances mentioned above are negligible in κ, essentially having the statistical distance be dealt with as part of the correctness slack. On the other hand, this can be avoided by letting $\mathcal{M}_{\mathsf{FDL}}$ be the ideal mechanism, thus leading to $(\varepsilon_\kappa, \delta_\kappa)$–SIM*-CDP where the statistical distance is rather incorporated into the δ_κ term. As having an ideal mechanism as close as possible to a standard SDP mechanism is to be seen as a more direct realisation of GED, we opt for having $\mathcal{M}_{RTGeo}$ as the ideal mechanism.

As stated in the preliminaries, we consider two-party computation schemes that operate in $\mathbb{F}_q$ with q being either a prime larger than 2 or a power of 2. We elaborate on active secure schemes for both domains in Appendix E. Implementing the FDL algorithm in either domain comes at a significant cost. Note that the Ber procedure and the first 2 steps of the FDL procedure consist of only binary arithmetics. However, the remainder of the FDL procedure consists of integer arithmetics. While there are protocols to evaluate the

[13] We also note that the requirement that $e^{-1/\lambda} < 1$ is equivalent to $\lambda > 0$, which is already guaranteed by $\lambda \in \mathbb{N}^{-1}$.

binary steps in the arithmetic domain, they are usually very costly. On the other hand, evaluating the whole algorithm in the binary domain comes with two problems: the summation and addition in binary would incur a significant cost, and second, the result would be a shared noise in the binary domain. Thus, applying the noise is limited to the binary domain. The mixed circuit approach (see Section 2.3) gives us a well-performing trade-off.

We accept inputs represented in the binary domain, perform all operations until the fourth step through a binary circuit, translate all shares to the arithmetic domain, and perform the rest of the operations through an arithmetic circuit. For each of these "phases", we use protocols introduced before. We use SPDZ$_{2^k}$ [20] for the arithmetic computations, the FKOS protocol [34] for binary circuits and *daBits* (doubly-authenticated bits) [66] for translating between the domains. With correct parametrization, we can achieve the same security guarantees in different computation domains. Thus, the feasibility of the mixed circuit approach is easily tested. The mixed circuit approach is feasible if switching between circuits is cheaper than the computation overhead in either domain. In our application (Section 6.1), we will, as typically for DP applications, focus on arithmetic computations. Evaluating the FDL mechanism in the binary domain would, therefore, incur a cost that scales with the underlying application. For the arithmetic case, we have an additional cost of assuring all input ranges (e.g., assert that binary coins $\in \{0, 1\}$) and evaluate binary gates with arithmetic circuits. Section 7 has a longer discussion about input validation.

We describe our protocol using the *Arithmetic Black Box (ABB)*, which is an ideal functionality in the UC framework. Very roughly, the ABB is a functionality that can take inputs from the parties and compute linear combinations and multiplications between stored values and output stored values. We use a flavor of the ABB that can do these operations over $\mathbb{F}_{2^k}$ and $\mathbb{F}_q$. Additionally, the ABB can translate values stored as elements of the binary field to binary values within the larger field. More concretely, we use the formulation of the ABB that can be found in [33] and we include a definition of the ideal functionality in Appendix B.1. Our protocol is presented in Figure 4.

We are now ready to present our main theorem, namely that the protocol we have introduced indeed is $(\varepsilon_\kappa, 0)$-SIM*-CDP. Let $decomp(\lambda, d)$ be short for the bit-decomposition of λ truncated to d bits. The proof is found in Appendix D.7.

Theorem 6.1. *Let $q > 2f_\kappa^{max} + 2B_\kappa$, $B_\kappa \in \{1, \ldots, \lceil q/2 \rceil - 1\}$, $\lambda_\kappa = \frac{\varepsilon_\kappa}{\Delta f_\kappa}$ and let $e^{-B_\kappa/\lambda_\kappa}$ and $B_\kappa 2^{-d_\kappa}$ be negligible in κ. Let $\{f_\kappa : \mathbb{Z}_q^{2N} \to \mathbb{Z}_q\}_{\kappa\in\mathbb{N}}$ be an ensemble of efficiently computable deterministic functions with $f_\kappa^{max} := \max_{D\in\mathbb{Z}_q^{2N}} |f_\kappa(D)|$. Let $\{\hat{f}_\kappa(D)\}_{\kappa\in\mathbb{N}}$ be $\{\mathcal{M}_{RTGeo}^{q,f_\kappa,\lambda_\kappa}(D)\}_{\kappa\in\mathbb{N}}$.*

Then $\pi_{\mathcal{M}_{\mathsf{FDL}}}(B_\kappa, d_\kappa, q, N, decomp(\lambda_\kappa, d_\kappa), f_\kappa)$ is an $(\varepsilon_\kappa, 0)$-SIM*-CDP protocol for the ideal functionality $\mathcal{F}_{SFE}^{\hat{f}_\kappa}$, with respect to the same adjacency notion as in the calculation of Δf_κ, in the $\mathcal{F}_{ABB}$-hybrid world.

Asymptotic computational cost. We consider the computational cost of $\pi_{\mathcal{M}_{\mathsf{FDL}}}$ in terms of calls to the ABB, ignoring the cost of computing f. This rough model for calculating computation cost is reasonable in two ways: Firstly, local operations are canonically negligible in terms of computation cost compared to operations that require interaction. Secondly, in practice, the instantiation of the ABB greatly influences the computation cost in practical terms. As is shown in EIKN [30], the asymptotic computational complexity of the FDL function is $O(Bd)$. This complexity follows directly from Definition 5.5 since all steps of the FDL procedure are repeated B times (that is, B Bernoulli trials are sampled and there are B elements in the sum) and within the Bernoulli trial subprocedure, all steps consist of d arithmetic operations.

It is important to note that the cost of sampling the noise is independent of the data query f. Relative DP usefulness intuitively increases as the number of elements in the input dataset grows. However, the performance of the sampling protocol scales with the number of queries and not with the size of the input dataset, thus amortizing its execution time further.

Protocol $\pi_{\mathcal{M}_{\mathsf{FDL}}}$

Parameters: Natural numbers B, d, q, N, bit decomposition $\hat{\alpha}^1, ..., \hat{\alpha}^d$ and an efficiently computable function $f : \mathbb{Z}_q^{2N} \to \mathbb{Z}_q$, meaning it can be computed using polynomially many multiplications and linear combinations in $\mathbb{Z}_q$. Assume access to $\mathcal{F}_{ABB}$.

Initialisation:

(1) Player i locally samples $Bd + 1$ fair coins and stores them as $\mathbf{e}^i$.
(2) Player i sends random seed vector $\mathbf{e}^i \in \mathbb{Z}_2^{Bd+1}$ as $Bd + 1$ consecutive inputs to $\mathcal{F}_{ABB}$ to be stored as elements of the binary field.
(3) For $j = 1, ..., Bd + 1$ the players compute $r_i \leftarrow e_j^1 \oplus e_j^2$ via $\mathcal{F}_{ABB}$.

Noise sampling:

(1) Each operation in the first two steps of the FDL specification is performed via $\mathcal{F}_{ABB}$. This results in the binary values $c_1, ..., c_B$ being computed as prefix-OR's of the Bernoulli trials.
(2) In $\mathcal{F}_{ABB}$, the values $c_1, ..., c_B$ and r_{Bd+1} are transformed to elements in the arithmetic field.
(3) All remaining operations in the FDL specification are performed via $\mathcal{F}_{ABB}$.

Finishing:

(1) Player 1 sends $\mathbf{x} \in \mathbb{Z}_q^N$ and player 2 sends $\mathbf{y} \in \mathbb{Z}_q^N$ to $\mathcal{F}_{ABB}$ and then f is computed via $\mathcal{F}_{ABB}$ according to its specification. The result is stored as $\bar{f}$.
(2) The sum of $\bar{f}$ and the FDL sample is computed via $\mathcal{F}_{ABB}$ and the result is output to the players.

Figure 4: The protocol description for the FDL mechanism in the $\mathcal{F}_{ABB}$-hybrid world.

6.1 Application: Integer Inner-products with Bounded Elements

We now compute integer inner-products using the $\pi_{\mathcal{M}_{FDL}}$ protocol. This query type is particularly interesting for a few reasons. First, it is non-linear and cannot be expressed as an aggregate function without knowledge of the other party's inputs. Second, it is a fundamental building block for more complicated queries like matrix

multiplications with vast applications in data processing, such as machine learning. To use $\pi_{\mathcal{M}_{FDL}}$, the query needs a bounded maximal absolute value, and for accuracy, we want the sensitivity of the query to be small. Therefore, we consider only inner-products where the input vectors have elements between $a \in \mathbb{Z}_q$ and $b \in \mathbb{Z}_q$.

Our setting provides security in the presence of active adversaries. Since these parties can deviate arbitrarily from the protocol, they might send inputs violating the above bounds. It is, therefore, necessary to prove the correctness of the input domain in both the FDL mechanism and the query function. There are different strategies to achieve such a feat. We note that the ABB accepts inputs of two types, either elements in the binary field or the larger finite field. We need to restrict the values to the pre-defined range for inputs in the arithmetic domain. Were we not to perform such an input validation, this would result in an increased sensitivity of the function (in relationship to what is a priori agreed upon by the two parties), thwarting the privacy level of the DP mechanism. In the presence of passive adversaries, however, there is of course no need to validate the inputs since the adversary will per definition not give out-of-range inputs. This requirement of a *proof of function sensitivity* also arises in other scenarios where the sensitivity is directly dependent on the secret data of multiple parties.

To provide such a validation that all given inputs fall within their allowed range, we consider two main options: Firstly, one could accept the inputs as elements in the larger field and then perform a zero-knowledge range proof[14] within the MPC domain. Alternatively, one could accept the inputs bit-by-bit and re-compose those bits into elements of the larger field. These approaches present a trade-off in input size and proof complexity. In the first approach, the cost of inputting a value is constant (i.e., depending on 2^k in our example) while proving the range is linear in the bound. In the bit-by-bit setting, the input and proving costs are both logarithmic in the bound. The second approach is thus more efficient for larger bound values depending on the specific scheme. As noted before, we opt for using the second method and we further assume that the difference between a and b is a power of 2, to facilitate inserting an input as a sequence of bits.

We consider DP with the bounded (*'change-one'*) adjacency notion and the data universe is $([a, b])^*$, such that each input D to f (as well as the protocol and the mechanism) is a tuple of $2N$ elements from $[a, b]$. Let $D := \mathbf{x}||\mathbf{y}$, i.e. the concatenation of the input vectors of the two parties. The inner-product $f(D)$ is defined as $\langle \mathbf{x}, \mathbf{y} \rangle := \sum_{i=1}^{N} x_i y_i$ with operations over $\mathbb{Z}_q$. The sensitivity Δf of the inner-product is $\max(|a^2 - ab|, |b^2 - ab|)$, under the assumption that $|f(\mathbf{x}||\mathbf{y})|$ is smaller than $\lfloor q/2 \rfloor$ such that field operations mimic integer behavior. We also have that $f^{max} = N \cdot \max(a^2, b^2)$.

Parameter choices. From the properties above, the following parameter considerations follow: Let the security parameter be the bit-length of a field element, i.e. $\kappa = \lceil \log_2(q) \rceil$, as is canonical. Let both ε_κ and Δf (by choice of a, b) be independent of κ. Further, we can set the FDL specific parameters as $B = d = \kappa$. Finally, we have $q > 2f^{max} + 2B = 2N \cdot \max(a^2, b^2) + 2B$, where the inequality holds for sufficiently large κ.

[14] For instance, such as described in the Bulletproofs paper [8].

In practice, one strategy is to choose κ as a canonical value for statistical security in cryptography, e.g., $\kappa = 40$, and then let this also be B and d.[15] The practical choice of ε is highly challenging, and there is a lively discussion in the literature on it, although consensus is largely lacking [27, 54, 55, 62]. Luckily, there is no direct dependence on the choice of ε in the other parameters. Finally, this leaves the choices of a, b, and N. Here, we care about the distance $|a - b|$ and the size of N. Both parameters allow for wider usage scenarios when increased. However, increasing N has adverse effects on runtime, and a larger distance causes a higher sensitivity and decreased usefulness (if ε is kept fixed). Finally, there is a trade-off between N and the sizes of a, b due to their dependence on q. In practice, this can be circumvented by increasing the modulus size q in the underlying MPC instantiation.

7 Implementation and Practical Performance

We tested our protocol by implementing it in the multi-protocol SPDZ (MP-SPDZ) [51] library. Among others, it provides efficient implementations of the SPDZ_{2^k} [20] and the FKOS [34] MPC schemes, and da-bit [66] and eda-bit [33] implementations. We implement procedure `Ber` in the FKOS scheme and procedure `FDL` in the mixed-circuit setting with FKOS and SPDZ_{2^k}. We find that only one switch between computation domains is necessary, making mixed-circuit computation highly competitive in performance. More precisely, this approach is faster than previous instantiations if the conversion cost is lower than the additional overhead in the unfit computation domain. Given the protocol in EIKN [30], circuit conversion has to be faster than the overhead of computing the Bernoulli and prefix-or functionality in the arithmetic domain.

In MPC schemes, communication is typically the bottleneck of efficient function evaluation. While some communication is necessary during the computation, much of the data transfer happens in a pre-processing phase. In our setup, we have three main components that require expensive pre-processing: shared randomness for inputs, authenticated multiplication triples, and doubly authenticated bits. In our inner-product use case, we only generate one `FDL` sample. However, most pre-processing operations come in blocks of size B or d. In our implementation, we take special care to minimize the communication rounds and adapt the pre-processing batch sizes to accommodate our protocol execution.

7.1 Benchmarks

In this section, we present benchmarks of our `FDL` mechanism with $B = d = \kappa$ and measure performance for different settings[16]. Relevant for parameter $\hat{\alpha}$, the bit decomposition of the Bernoulli bias, is the decomposition length d. When setting a value α, the binary decomposition truncates this value to the predefined precision. Although our code can be instantiated with any number of parties, we fixed the number of parties to 2 as to align with the formalities of earlier sections. We provide exemplary data points at 40- and 80-bit, typical statistical security parameters. Next, we evaluate the mechanism at 128-bit, a usual conservative choice as a computational

[15] Note that we use κ as a *computational* security parameter but that statistical security implies computational security. One appealing alternative is to introduce an additional statistical parameter separate from κ, let them be proportional and align B, d to the new parameter instead.

[16] The code can be found at https://github.com/Fable95/laplace_sampler.

security parameter. Note that, in MP-SPDZ, the underlying security parameters for SPDZ_{2^k} are fixed to 64-bit computational and 64-bit statistical security. We run all benchmarks on a Linux server with an AMD Ryzen 9 7900X CPU (4.7 GHz). Each party only has access to one thread for computations. We separate our results into single sample computation and amortized evaluation for 1000 samples. The single sample evaluation is further split into the pre-processing and online phases of MPC, where the pre-processing step consists of generating necessary multiplication triples and da-bits.

Table 1 presents the runtime metrics for different network settings. In Setting 1, we have an unrestricted LAN setup. Setting 2 simulates a less powerful LAN setup by limiting the network to 1Gbit/s and the round-trip time (RTT) to 1ms. Finally, in Setting 3, we simulate a WAN network with 100Mbit/s and 100ms RTT, reflecting a solid but distant connection (e.g., intercontinental). Given the asymptotic complexity $O(Bd)$, the runtime results reflect the expected quadratic growth in the security parameter. Regarding the network settings, communication is needed for inputs, binary AND gates, arithmetic multiplication, secret share conversion, and outputs. Since inputs, conversions, and computations depend on one or both parameters B, or d, the negative impact of a reduced network speed and increased RTT is increased. Compared to concurrent work [50], our mechanism outperforms theirs in runtime and memory for the overall computation in the fast network settings.[17] Arguably, their setup heavily optimizes the online phase, making it more efficient if pre-processing can be performed in advance. However, sampling geometric noise in MPC can generally be seen as pre-processing since the sensitivity of a function is known before the data is processed, and the parties can already engage in the noise sampling procedure before their inputs to the query function have been fixed. Further, their geometric mechanism has a low round complexity, showing improved performance in WAN network settings. Comparing with [30] is challenging as only asymptotic complexities are given there and the results are based on arithmetic evaluations of binary computations from [65]. Our approach, on the other hand, is based on mixed circuits [66] and includes substantial performance improvements by dedicated parameter optimizations. In our benchmarks, we adhered to the following principle. We aimed to reduce the communication complexity for low-latency networks, while for the high-latency networks, we reduced the round complexity. This trade-off can be determined with the pre-processing batch size parameter. Given a high minimum batch size for the MPC schemes we use, computing a single geometric sample leads to substantial overhead. Thus, it is crucial to parametrize the implementation according to the number of samples and expected network latency.

In Table 2, we present benchmarks for network costs for each security parameter. We see that the network cost of our implementation is lower than that in [50], further showing that their round complexity is much lower than that of the malicious secure SPDZ_{2^k} protocol. Given the network cost, we could further reduce the network bandwidth before its limiting impact equals a slow RTT. In our amortized costs column, we present the network traffic per sample in a computation of 1 000 samples.

[17] One should further note that [50] is in the more efficient setting of passive adversaries, thus making direct comparisons skewed in their favor.

Table 1: Runtime in milliseconds of benchmarks with different security levels. Total runtime is for a single sample, while amortized runtime assumes 1000 samples.

Protocol	κ	Prep.	Online	Total	Amort.
10 Gbit/s with RTT of 1 ms					
Ours	40	74.7	42	116.6	34.6
	80	94.2	119.9	214.1	118.5
	128	130	276.9	406.9	283.4
[50]	40	1606	37.72	1 643	992†
1 Gbit/s with RTT of 1 ms					
Ours	40	182.9	248.4	431.2	69.7
	80	245.6	650.2	895.7	209.7
	128	345.6	1 362	1 707	520.3
[50]	40	4 707	4.81	–	4 711‡
100 Mbit/s with RTT of 100 ms					
Ours	40	11 256	20 486	31 742	577.9
	80	15 215	51 794	67 009	1 604
	128	20 795	105 350	126 145	3 558
[50]	40	42 352	47.99	–	42 400‡

† Amortized over 40 samples

‡ Amortized over 10 samples, no single sample performance provided.

Table 2: Network cost in MB of different geometric sampling settings. The amortized cost assumes 1000 samples.

Protocol	κ	Prep.	Online	Total	Amort.
Ours	40	14.7	17.9	65.3	23.8
	80	20.9	58.3	158.3	75.3
	128	29.2	143.4	345.2	173.6
[50]	40	–	–	492.7†	–

† Run with single sample, no amortized network cost provided.

8 Conclusion and Outlooks

In this work, we revisit the idea of generic emulation of the central dataholder (GED) as a method to achieve accuracy equal to that of the central model of DP without the need for a single trusted dataholder. The bulk of our work is spent analysing existing definitions of computational DP (CDP) in the multiparty setting, noting that whereas they are very well-suited for theoretic study and use with special-purpose MPC schemes, they all fit somewhat suboptimally to the task of GED. Since one of them, SIM^{+}-CDP, appears to fit very well conceptually but has some details preventing its use together with canonical statistical DP (SDP) mechanisms, we propose both a generalised version of it, SIM°-CDP, and another instantiation of that generalised definition, SIM^{*}-CDP, that we argue is more fitting to the current state-of-the-art in both general-purpose MPC and SDP. We relate SIM^{*}-CDP to IND-CDP and SIM-CDP, showing that it is a stronger notion in the sense that all SIM^{*}-CDP protocols

are also SIM-CDP (and thus also IND-CDP) with unchanged parameters, whilst there are computational tasks (such as computing a given functionality to within a given absolute error with constant probability) that can be solved with SIM-CDP but are impossible with SIM*-CDP. Further, we show that SIM$^+$-CDP and SIM*-CDP are separated from each other in both directions in the sense that there are tasks solvable for one of them but not the other. Some of these results, however, are established under specific parameter regimes (as is commonplace in the CDP literature) and therefore extending them to wider regimes is an interesting open problem. On the practical side, we show how to achieve SIM*-CDP via the truncated geometric (discrete Laplace) mechanism by using a state-of-the-art protocol for distributed noise sampling and analyse the use of this protocol for computing integer inner-products with active security in the two-party setting. We then provide an open-sourced implementation of the protocol using the MP-SPDZ library and show that it is very efficient in practice.

As always when formulating new definitions in cryptography questions arise, such as whether the definition is intuitive, practically usable, and not overly relaxed or strict. On the usability front, we present evidence that SIM*-CDP is practical since it allows us to design efficient, quite general protocols of natural tasks that fulfill it, and the proof that the definition is satisfied follows essentially directly from the use of general-purpose MPC and an SDP mechanism. Further, the definition appears intuitive due to its closeness to both previous definitions and established formalities in both the DP and MPC domains. There is, however, much need for additional scrutiny, and this is the case also for the question about balance in the definition. Interesting open directions here are to more tightly relate the definition to previous ones and explore whether there is some characteristic trait of SDP that is captured in the previous ones but not in SIM*-CDP.

Acknowledgments

We gratefully thank the authors of MPRV [64] for providing a full version of the paper and for answering our questions. We thank anonymous PoPETS reviewers for their comments and we thank Lea Demelius for reading early drafts and giving feedback that improved the disposition significantly. This work is partly supported by the European Union under the project Confidential6G with Grant agreement ID: 101096435.

References

[1] Balamurugan Anandan and Chris Clifton. 2015. Laplace noise generation for two-party computational differential privacy. In *2015 13th Annual Conference on Privacy, Security and Trust (PST)*. 54–61. https://doi.org/10.1109/PST.2015.7232954

[2] Victor Balcer and Salil Vadhan. 2019. Differential Privacy on Finite Computers. *Journal of Privacy and Confidentiality* 9, 2 (Sep. 2019). https://doi.org/10.29012/jpc.679

[3] Donald Beaver. 1996. Correlated Pseudorandomness and the Complexity of Private Computations. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing* (Philadelphia, Pennsylvania, USA) *(STOC '96)*. Association for Computing Machinery, New York, NY, USA, 479–488. https://doi.org/10.1145/237814.237996

[4] Amos Beimel, Kobbi Nissim, and Eran Omri. 2008. Distributed Private Data Analysis: Simultaneously Solving How and What. In *Advances in Cryptology – CRYPTO 2008*, David Wagner (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 451–468.

[5] James Bell, Adrià Gascón, Badih Ghazi, Ravi Kumar, Pasin Manurangsi, Mariana Raykova, and Phillipp Schoppmann. 2022. Distributed, Private, Sparse Histograms in the Two-Server Model *(CCS '22)*. Association for Computing Machinery, New York, NY, USA, 307–321. https://doi.org/10.1145/3548606.3559383

[6] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. 2017. Prochlo. In *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM. https://doi.org/10.1145/3132747.3132769

[7] Mark Bun, Yi-Hsiu Chen, and Salil Vadhan. 2016. Separating Computational and Statistical Differential Privacy in the Client-Server Model. In *Theory of Cryptography*, Martin Hirt and Adam Smith (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 607–634.

[8] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. 2018. Bulletproofs: Short Proofs for Confidential Transactions and More. In *2018 IEEE Symposium on Security and Privacy (SP)*. 315–334. https://doi.org/10.1109/SP.2018.00020

[9] Ran Canetti. 2000. Security and Composition of Multiparty Cryptographic Protocols. *J. Cryptol.* 13, 1 (jan 2000), 143–202. https://doi.org/10.1007/s001459910006

[10] Ran Canetti. 2000. Universally Composable Security: A New Paradigm for Cryptographic Protocols. https://eprint.iacr.org/2000/067

[11] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. 2003. On the limitations of universally composable two-party computation without set-up assumptions. In *Proceedings of the 22nd International Conference on Theory and Applications of Cryptographic Techniques* (Warsaw, Poland) *(EUROCRYPT'03)*. Springer-Verlag, Berlin, Heidelberg, 68–86.

[12] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. 2002. Universally Composable Two-Party and Multi-Party Secure Computation. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing* (Montreal, Quebec, Canada) *(STOC '02)*. Association for Computing Machinery, New York, NY, USA, 494–503. https://doi.org/10.1145/509907.509980

[13] Clement Canonne, Gautam Kamath, and Thomas Steinke. 2022. The Discrete Gaussian for Differential Privacy. *Journal of Privacy and Confidentiality* 12, 1 (Jul. 2022). https://doi.org/10.29012/jpc.784

[14] Jeffrey Champion, abhi shelat, and Jonathan Ullman. 2019. Securely Sampling Biased Coins with Applications to Differential Privacy. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (London, United Kingdom) *(CCS '19)*. Association for Computing Machinery, New York, NY, USA, 603–614. https://doi.org/10.1145/3319535.3354256

[15] T-H. Hubert Chan, Elaine Shi, and Dawn Song. 2012. Optimal Lower Bound for Differentially Private Multi-party Aggregation. In *Algorithms – ESA 2012*, Leah Epstein and Paolo Ferragina (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 277–288.

[16] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. 2019. Distributed Differential Privacy via Shuffling. In *Advances in Cryptology – EUROCRYPT 2019*, Yuval Ishai and Vincent Rijmen (Eds.). Springer International Publishing, Cham, 375–403.

[17] Albert Cheu and Chao Yan. 2023. Necessary Conditions in Multi-Server Differential Privacy. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 251)*, Yael Tauman Kalai (Ed.). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 36:1–36:21. https://doi.org/10.4230/LIPIcs.ITCS.2023.36

[18] B. Chor and E. Kushilevitz. 1989. A zero-one law for Boolean privacy. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing* (Seattle, Washington, USA) *(STOC '89)*. Association for Computing Machinery, New York, NY, USA, 62–72. https://doi.org/10.1145/73007.73013

[19] Henry Corrigan-Gibbs and Dan Boneh. 2017. Prio: Private, Robust, and Scalable Computation of Aggregate Statistics. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. USENIX Association, Boston, MA, 259–282. https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/corrigan-gibbs

[20] Ronald Cramer, Ivan Damgård, Daniel Escudero, Peter Scholl, and Chaoping Xing. 2018. SPD$\mathbb{Z}_{2^k}$: Efficient MPC mod 2^k for Dishonest Majority. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 10992)*, Hovav Shacham and Alexandra Boldyreva (Eds.). Springer, 769–798. https://doi.org/10.1007/978-3-319-96881-0_26

[21] Ronald Cramer, Ivan Bjerre Damgård, and Jesper Buus Nielsen. 2015. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press. https://doi.org/10.1017/CBO9781107337756

[22] Ivan Damgård and Jesper Buus Nielsen. 2003. Universally Composable Efficient Multiparty Computation from Threshold Homomorphic Encryption. In *Advances in Cryptology - CRYPTO 2003*, Dan Boneh (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 247–264.

[23] Ivan Damgård, Valerio Pastro, Nigel Smart, and Sarah Zakarias. 2012. Multiparty Computation from Somewhat Homomorphic Encryption. In *Advances in Cryptology – CRYPTO 2012*, Reihaneh Safavi-Naini and Ran Canetti (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 643–662.

[24] Daniel Demmler, Thomas Schneider, and Michael Zohner. 2015. ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation.

In *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*. The Internet Society. https://www.ndss-symposium.org/ndss2015/aby---framework-efficient-mixed-protocol-secure-two-party-computation

[25] Cynthia Dwork. 2006. Differential privacy. In *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II* (Venice, Italy) *(ICALP'06)*. Springer-Verlag, Berlin, Heidelberg, 1–12. https://doi.org/10.1007/11787006_1

[26] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our Data, Ourselves: Privacy via Distributed Noise Generation. In *Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques* (St. Petersburg, Russia) *(EUROCRYPT'06)*. Springer-Verlag, Berlin, Heidelberg, 486–503. https://doi.org/10.1007/11761679_29

[27] Cynthia Dwork, Nitin Kohli, and Deirdre Mulligan. 2019. Differential Privacy in Practice: Expose your Epsilons! *Journal of Privacy and Confidentiality* 9 (10 2019). https://doi.org/10.29012/jpc.689

[28] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. *Theory of Cryptography* Vol. 3876, 265–284. https://doi.org/10.1007/11681878_14

[29] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.

[30] Reo Eriguchi, Atsunori Ichikawa, Noboru Kunihiro, and Koji Nuida. 2021. Efficient Noise Generation to Achieve Differential Privacy with Applications to Secure Multiparty Computation. In *Financial Cryptography and Data Security: 25th International Conference, FC 2021, Virtual Event, March 1–5, 2021, Revised Selected Papers, Part I*. Springer-Verlag, Berlin, Heidelberg, 271–290. https://doi.org/10.1007/978-3-662-64322-8_13

[31] Reo Eriguchi, Atsunori Ichikawa, Noboru Kunihiro, and Koji Nuida. 2023. Efficient Noise Generation Protocols for Differentially Private Multiparty Computation. *IEEE Transactions on Dependable and Secure Computing* 20, 6 (2023), 4486–4501. https://doi.org/10.1109/TDSC.2022.3227568

[32] Daniel Escudero. 2022. An Introduction to Secret-Sharing-Based Secure Multiparty Computation. https://eprint.iacr.org/2022/062

[33] Daniel Escudero, Satrajit Ghosh, Marcel Keller, Rahul Rachuri, and Peter Scholl. 2020. Improved Primitives for MPC over Mixed Arithmetic-Binary Circuits. In *Advances in Cryptology – CRYPTO 2020*, Daniele Micciancio and Thomas Ristenpart (Eds.). Springer International Publishing, Cham, 823–852.

[34] Tore Kasper Frederiksen, Marcel Keller, Emmanuela Orsini, and Peter Scholl. 2015. A Unified Approach to MPC with Preprocessing Using OT. In *Proceedings, Part I, of the 21st International Conference on Advances in Cryptology – ASIACRYPT 2015 - Volume 9452*. Springer-Verlag, Berlin, Heidelberg, 711–735. https://doi.org/10.1007/978-3-662-48797-6_29

[35] Badih Ghazi, Rahul Ilango, Pritish Kamath, Ravi Kumar, and Pasin Manurangsi. 2023. Towards Separating Computational and Statistical Differential Privacy. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*. 580–599. https://doi.org/10.1109/FOCS57990.2023.00042

[36] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. 2012. Universally Utility-maximizing Privacy Mechanisms. *SIAM J. Comput.* 41, 6 (2012), 1673–1693. https://doi.org/10.1137/09076828X

[37] Oded Goldreich. 2004. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, USA.

[38] Oded Goldreich, Shafi Goldwasser, and Asaf Nussboim. 2010. On the Implementation of Huge Random Objects. *SIAM J. Comput.* 39, 7 (2010), 2761–2822. https://doi.org/10.1137/080722771

[39] Vipul Goyal, Dakshita Khurana, Ilya Mironov, Omkant Pandey, and Amit Sahai. 2016. Do Distributed Differentially-Private Protocols Require Oblivious Transferl. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 55)*, Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi (Eds.). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 29:1–29:15. https://doi.org/10.4230/LIPIcs.ICALP.2016.29

[40] Vipul Goyal, Ilya Mironov, Omkant Pandey, and Amit Sahai. 2013. Accuracy-Privacy Tradeoffs for Two-Party Differentially Private Protocols. In *Advances in Cryptology – CRYPTO 2013*, Ran Canetti and Juan A. Garay (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 298–315.

[41] Adam Groce, Jonathan Katz, and Arkady Yerukhimovich. 2011. Limits of Computational Differential Privacy in the Client/Server Setting. In *Theory of Cryptography*, Yuval Ishai (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 417–431.

[42] Adam Groce, Peter Rindal, and Mike Rosulek. 2019. Cheaper Private Set Intersection via Differentially Private Leakage. *Proceedings on Privacy Enhancing Technologies* 2019 (07 2019), 6–25. https://doi.org/10.2478/popets-2019-0034

[43] Iftach Haitner, Noam Mazor, Ronen Shaltiel, and Jad Silbak. 2019. Channels of Small Log-Ratio Leakage and Characterization of Two-Party Differentially Private Computation. In *Theory of Cryptography*, Dennis Hofheinz and Alon Rosen (Eds.). Springer International Publishing, Cham, 531–560.

[44] Iftach Haitner, Noam Mazor, Jad Silbak, and Eliad Tsfadia. 2022. On the Complexity of Two-Party Differential Privacy. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing* (Rome, Italy) *(STOC 2022)*. Association for Computing Machinery, New York, NY, USA, 1392–1405. https://doi.org/10.1145/3519935.3519982

[45] Xi He, Ashwin Machanavajjhala, Cheryl Flynn, and Divesh Srivastava. 2017. 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* (Dallas, Texas, USA) *(CCS '17)*. Association for Computing Machinery, New York, NY, USA, 1389–1406. https://doi.org/10.1145/3133956.3134030

[46] Martin Hirt, Ueli Maurer, and Vassilis Zikas. 2008. MPC vs. SFE: Unconditional and Computational Security. In *Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology* (Melbourne, Australia) *(ASIACRYPT '08)*. Springer-Verlag, Berlin, Heidelberg, 1–18. https://doi.org/10.1007/978-3-540-89255-7_1

[47] Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. 2006. On combining privacy with guaranteed output delivery in secure multiparty computation. In *Proceedings of the 26th Annual International Conference on Advances in Cryptology* (Santa Barbara, California) *(CRYPTO'06)*. Springer-Verlag, Berlin, Heidelberg, 483–500. https://doi.org/10.1007/11818175_29

[48] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2011. What Can We Learn Privately? *SIAM J. Comput.* 40, 3 (2011), 793–826. https://doi.org/10.1137/090756090

[49] Dana Keeler, Chelsea Komlo, Emily Lepert, Shannon Veitch, and Xi He. 2023. DPrio: Efficient Differential Privacy with High Utility for Prio. *Proceedings on Privacy Enhancing Technologies* (2023).

[50] Hannah Keller, Helen Möllering, Thomas Schneider, Oleksandr Tkachenko, and Liang Zhao. 2024. Secure Noise Sampling for DP in MPC with Finite Precision. In *Proceedings of the 19th International Conference on Availability, Reliability and Security* (Vienna, Austria) *(ARES '24)*. Association for Computing Machinery, New York, NY, USA, Article 25, 12 pages. https://doi.org/10.1145/3664476.3664490

[51] Marcel Keller. 2020. MP-SPDZ: A Versatile Framework for Multi-Party Computation. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (Virtual Event, USA) *(CCS '20)*. Association for Computing Machinery, New York, NY, USA, 1575–1590. https://doi.org/10.1145/3372297.3417872

[52] Marcel Keller, Emmanuela Orsini, and Peter Scholl. 2016. MASCOT: Faster Malicious Arithmetic Secure Computation with Oblivious Transfer. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (Vienna, Austria) *(CCS '16)*. Association for Computing Machinery, New York, NY, USA, 830–842. https://doi.org/10.1145/2976749.2978357

[53] Dakshita Khurana, Hemanta K. Maji, and Amit Sahai. 2014. Black-Box Separations for Differentially Private Protocols. In *Advances in Cryptology – ASIACRYPT 2014*, Palash Sarkar and Tetsu Iwata (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 386–405.

[54] Daniel Kifer, John M. Abowd, Robert Ashmead, Ryan Cumings-Menon, Philip Leclerc, Ashwin Machanavajjhala, William Sexton, and Pavel Zhuravlev. 2022. Bayesian and Frequentist Semantics for Common Variations of Differential Privacy: Applications to the 2020 Census. arXiv:2209.03310 [cs.CR]

[55] Sara Krehbiel. 2019. Choosing Epsilon for Privacy as a Service. *Proceedings on Privacy Enhancing Technologies* 2019 (2019), 192 – 205.

[56] Yehuda Lindell. 2017. *How to Simulate It – A Tutorial on the Simulation Proof Technique*. Springer International Publishing, Cham, 277–346.

[57] Yehuda Lindell. 2021. Secure Multiparty Computation. *Commun. ACM* 64, 1 (dec 2021), 86–96. https://doi.org/10.1145/3387108

[58] Helger Lipmaa and Tomas Toft. 2013. Secure Equality and Greater-Than Tests with Sublinear Online Complexity. In *Automata, Languages, and Programming*, Fedor V. Fomin, Rūsiņš Freivalds, Marta Kwiatkowska, and David Peleg (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 645–656.

[59] Sahar Mazloom and S. Dov Gordon. 2018. Secure Computation with Differentially Private Access Patterns. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (Toronto, Canada) *(CCS '18)*. Association for Computing Machinery, New York, NY, USA, 490–507. https://doi.org/10.1145/3243734.3243851

[60] Andrew McGregor, Ilya Mironov, Toniann Pitassi, Omer Reingold, Kunal Talwar, and Salil Vadhan. 2010. The Limits of Two-Party Differential Privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. 81–90. https://doi.org/10.1109/FOCS.2010.14

[61] Andrew Mcgregor, Ilya Mironov, Toniann Pitassi, Omer Reingold, Kunal Talwar, and Salil P. Vadhan. 2011. The Limits of Two-Party Differential Privacy. *Electronic Colloquium on Computational Complexity* 18 (2011), 106.

[62] Luise Mehner, Saskia Nuñez von Voigt, and Florian Tschorsch. 2021. Towards Explaining Epsilon: A Worst-Case Study of Differential Privacy Risks. *2021 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (2021), 328–331.

[63] Fredrik Meisingseth and Christian Rechberger. 2025. SoK: Computational and Distributed Differential Privacy for MPC. *Proceedings on Privacy Enhancing*

Technologies (2025), to appear. https://eprint.iacr.org/2024/1290
[64] Ilya Mironov, Omkant Pandey, Omer Reingold, and Salil Vadhan. 2009. Computational Differential Privacy. In *Advances in Cryptology - CRYPTO 2009*, Shai Halevi (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 126–142.
[65] Takashi Nishide and Kazuo Ohta. 2007. Multiparty Computation for Interval, Equality, and Comparison Without Bit-Decomposition Protocol. In *Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, April 16-20, 2007, Proceedings (Lecture Notes in Computer Science, Vol. 4450)*, Tatsuaki Okamoto and Xiaoyun Wang (Eds.). Springer, 343–360. https://doi.org/10.1007/978-3-540-71677-8_23
[66] Dragos Rotaru and Tim Wood. 2019. MArBled Circuits: Mixing Arithmetic and Boolean Circuits with Active Security. In *Progress in Cryptology – INDOCRYPT 2019: 20th International Conference on Cryptology in India, Hyderabad, India, December 15–18, 2019, Proceedings* (Hyderabad, India). Springer-Verlag, Berlin, Heidelberg, 227–249. https://doi.org/10.1007/978-3-030-35423-7_12
[67] Phillipp Schoppmann, Lennart Vogelsang, Adrià Gascón, and Borja Balle. 2020. Secure and Scalable Document Similarity on Distributed Databases: Differential Privacy to the Rescue. *Proceedings on Privacy Enhancing Technologies* 2020 (2020), 209 – 229.
[68] Salil Vadhan. 2017. *The Complexity of Differential Privacy.* Springer International Publishing, Cham, 347–450. https://doi.org/10.1007/978-3-319-57048-8_7
[69] Emanuele Viola. 2012. The Complexity of Distributions. *SIAM J. Comput.* 41, 1 (2012), 191–218. https://doi.org/10.1137/100814998
[70] Thomas W. Watson. 2013. *The Computational Complexity of Randomness.* Ph. D. Dissertation. https://www.proquest.com/dissertations-theses/computational-complexity-randomness/docview/1441711479/se-2
[71] Chengkun Wei, Ruijing Yu, Yuan Fan, Wenzhi Chen, and Tianhao Wang. 2023. Securely Sampling Discrete Gaussian Noise for Multi-Party Differential Privacy. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security* (, Copenhagen, Denmark,) *(CCS '23)*. Association for Computing Machinery, New York, NY, USA, 2262–2276. https://doi.org/10.1145/3576915.3616641
[72] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. *Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity.* 2468–2479. https://doi.org/10.1137/1.9781611975482.151

A The Efficient Samplability of Distributions

We consider a probability distribution Dist efficiently samplable if there exists a PPT Turing Machine (TM) that maps 1^κ to a sample from Dist. That is, if there exists a probabilistic TM that runs in strict polynomial time and, using only its unbiased randomness tape, samples from Dist exactly. We do not delve into the broader literature on what distributions can be sampled (exactly or approximately) under certain constraints (see, for instance [38, 69, 70]), but rather settle for some basic remarks. Most critically, we note that since the sampler runs in polynomial time, it can read at most $poly(\kappa)$ coins from its randomness tape, with $poly(\cdot)$ being some polynomial. Since the randomness tape is the only source of randomness in the sampler, the sampler is deterministic if one considers the coins as input. This means that there are only $2^{poly(\kappa)}$ possible executions of the sampler, each giving a fixed output.[18] This has two direct consequences:

- **All probability densities in** Dist **must be multiples of** $2^{-poly(\kappa)}$.
- **The support of** Dist **can contain at most** $2^{poly(\kappa)}$ **distinct elements.**

The first of these, of course, implies the second and the second can similarly be directly realised by that the sampler can write at most $poly(\kappa)$ elements on its output tape, since it is strict PPT. The restriction on the support is anyhow useful to include explicitly, since it implies that only discrete distributions on a sufficiently small support can be efficiently sampled. This rules out, most directly, sampling from the reals (as in the usual Laplace or Gaussian distributions) but also, a bit more subtly, sampling distributions whose support is of a finite size $q > 2^{poly(\kappa)}$. The first restriction however is the more important one, and in particular, it rules out distributions such as

- **Bernoulli trials of general parameters:** There are parameters α such that the Bernoulli distribution $Ber(\alpha)$ can not be efficiently sampled. One example of this is $\alpha = 1/3$. In particular, $Ber(\alpha)$ is efficiently samplable iff α is a multiple of $2^{-poly(\kappa)}$ for some polymial $poly(\cdot)$.
- **Truncated geometric and discrete gaussian distributions of general parameters:** There are parameters α such that the geometric distribution (discrete Laplace) on a finite (small) support $Geo(\alpha)$ can not be efficiently sampled. This is due to the need to generate probability densities of the form $\frac{1}{2\alpha} e^{\frac{-|z|}{\alpha}}$, which generally cannot be expressed as a multiple of $2^{-poly(\kappa)}$. The same reasoning holds for the discrete gaussian.

The takeaway from these examples, since we still would like to use these kinds of distributions when constructing DP mechanisms, is that one must either relax the demand for *strict* polynomial time or the demand that the samples are from *exactly* Dist rather than from a good approximation of Dist. Indeed, is is shown in [13] that the discrete gaussian (with support on the integers) can be sampled in *expected* polynomial time. In practice, settling for expected polynomial time is arguably not at all problematic, at least in the central model. The problems arise on the theory side when trying to prove security in a protocol, since the literature on secure computation heavily favours strict polynomial time, meaning that directly slotting in a mechanism that potentially runs, say, exponentially long might prove a large obstacle to proving security of the protocol as a whole. The other alternative is to settle for approximating the distribution in question, a strategy arguably more readily usable in conjunction with formally proving secure computation, as we do in this work. The challenge to this approach however is to include this sampling error into the DP guarantee in question, for instance letting it be a part of the δ parameter or including it in a computational error term in the CDP notion in question.

B The Ideal/real Paradigm, Standalone and UC security

We now give a brief introduction to the real/ideal-world paradigm of security and it's two standard versions, the *standalone security model* and the *universal composability (UC) security model*. Due to their complicated nature, we will not be able to describe them in full formal detail and we refer to [10, 21] for details on the UC model and to [9, 37, 56] for details on the standalone model. In our summary here, we lean upon those in [32, 42, 57].

The core idea of the ideal/real paradigm of security is to define an *ideal world* that is secure by definition, i.e. which formulates what computations are supposed to be done and what it means formally to have that done securely (for instance, specifying what types of information leakage are not to be seen as a violation of security). Then the security of the actual protocol in question, defining the *real world*, is asserted by a simulation proof that the adversary cannot know if it is interacting with the ideal world or the real

[18] This simple fact was made aware to us by [2, 13].

world. That is, the intuition is that if the adversary cannot tell if it is interacting with the real protocol or a version of the protocol that is secure by definition, then the protocol should be seen as essentially as secure as that in the ideal world.

The ideal world works as follows: There is an incorruptible third party called the *ideal functionality* which is given the inputs of all of the parties. The functionality then performs the computation in question, perhaps incorporating some well-defined allowed influence of the adversary, and then forwards the result to the players who output it. The functionality thereby defines what it means to be secure, and what computations should be possible, by merit of being incorruptible. However, when observing the protocol execution, it is potentially very easy to distinguish such an ideal world from the real world, for instance by observing the number of messages sent. Therefore, the ideal world also must include a *simulator*, or *ideal-world adversary*, whose task it is to generate a view that is indistinguishable from that of the real-world adversary and to do this by using only the information available to it in the ideal world (essentially, the information given to it by the ideal functionality). An important quantifier of the strength of the simulation argument is the efficiency of the simulator, since it describes how much work is needed to turn the allowed information leakage into the real one, with an efficient simulator giving a stronger guarantee of security. Therefore, in the literature on secure computation, one typically requires the simulator to be efficient, although this is not always the case for CDP using the ideal/real-world paradigm.[19]

So the core idea is that the ideal world (with parties, ideal functionality and simulator) in some sense looks like the real world (with parties and adversary). This begs the question of who they should look the same to – who is the distinguisher? Here is where the standalone and UC security models start diverging. In the standalone model, the distinguisher is the adversary, meaning that the distinguisher itself takes part in the protocol. More precisely, the task of the simulator is to use only information available in the ideal world and generate an output distribution that is indistinguishable from the view[20] of the real-world adversary.

Definition B.1 (Standalone security, reformulation of Def. 4 in [9]). We say that a protocol π is a secure protocol for the functionality $\mathcal{F}$ if for all efficient adversaries $\mathcal{A}$, there exists an efficient simulator $\mathcal{S}$ (corrupting the same parties as $\mathcal{A}$) such that the joint output of the honest parties and $\mathcal{A}$ in the real world is computationally indistinguishable from the joint output of the honest parties and $\mathcal{S}$ in the ideal world, i.e. when the output distributions in the ideal and real worlds are computationally indistinguishable.

There are many different flavors of the security definition, for instance in the type of indistinguishability (sometimes one requires the distributions to be identical or have negligible statistical distance) or in the inclusions of specific requirements on the *correctness* (such as requiring that the outputs in the real and ideal worlds are identical or statistically close if there are no corruptions, as done in [37, 56]). The version of the notion which is used in MPRV [64] within the definition of SIM$^+$-CDP (Definition 3.3) has such an extra correctness requirement, as well as demanding efficient protocols and removing the efficiency requirement of the simulator.

Definition B.2 (Standalone security as in MPRV [64], Reformulated). We say that a protocol π is a secure protocol for the functionality $\mathcal{F}$ if it fulfills Definition B.1 with the following changes:

(1) π must be *efficiently computable* (PPT);
(2) π must have *perfect correctness*, that is, in an honest execution of π, its output distribution is identical to that of $\mathcal{F}$;
(3) the simulator is allowed to be inefficient.

In the standalone model, as the name implies, the security of the protocol is considered in isolation, meaning that the distinguisher is constrained to what other protocols it can run in order to try and distinguish the worlds. Making such a restriction makes proving security technically easier, for example by allowing so-called rewinding techniques. The drawback of the model is precisely that it considers protocol security in isolation, opening up the possibility that a protocol thought to be secure loses all of its security properties when it is run in parallel to some other processes. Since it can be argued that such composition of protocols and processes is the rule rather than the exception in modern computer systems, it is highly desirable to be able to prove that a protocol remains secure also when other protocols are run in composition to it.

There are many ways to compose protocols and some of them are easier to deal with than others. For instance, the usual formulations of the standalone model guarantee that security is preserved under *sequential composition*, meaning as long as all the surrounding protocols are run sequentially (one after another). The most powerful type of composition results are those when the security of the protocol is preserved regardless of how the surrounding protocols are run (in particular when they run concurrently to the protocol in question). This is called *universal composition* and the entire point of the UC (Universal Composability) security framework is that protocols proven within it remain secure under universal composition. This means, in particular, that if a protocol π realises the ideal functionality $\mathcal{F}$, then any other protocol that uses $\mathcal{F}$ as a subprocedure does not lose its security properties if $\mathcal{F}$ is replaced by a copy of π. In the UC framework, the distinguisher goes from taking part in the protocol (as in the standalone model) to being an external entity that observes and interacts with the system from the outside. The distinguisher is captured in an entity called *the environment*, which is an entity in both worlds that selects the initial inputs to all parties, interacts arbitrarily with the adversary and then, based on the outputs of each party at the end, tries to distinguish between the two worlds. In other words, the environment gets to play with one of the worlds and depending only on the input-output behaviour of this world it tries to determine which world it is playing with.

Definition B.3 (UC security [10, 42]). We say that an efficient protocol π UC-securely realises the ideal functionality $\mathcal{F}$ if for all efficient real-world adversaries $\mathcal{A}$ there exists an efficient simulator[21] $\mathcal{S}$ (corrupting the same parties as $\mathcal{A}$) such that for all efficient environments E, the statistical distance between E's output when interacting with the ideal world and that when interacting with the real world is negligible in the security parameter κ.

[19] Most notably, in MPRV [64], the simulators are allowed to be inefficient (computationally unbounded) in the definition of SIM$^+$-CDP.

[20] In Definition B.1 it is the output rather than the view of the adversary that is considered. These two formulations are equivalent since the adversary is allowed to simply output its entire view as output.

[21] Also called *ideal-world adversary*.

B.1 The Arithmetic Black-box

In Figure 5 we present the ideal functionality $\mathcal{F}_{ABB}$ of the arithmetic black-box. The ABB is at times formulated slightly differently, such as only operating within the arithmetic domain, not including conversions between the domains or including conversions in both directions in between the binary and arithmetic representations. We choose the flavor of ABB that is used in [33], simply because it includes the operations we need but nothing more. For more details on the ABB see, for instance, [22, 58].

Functionality $\mathcal{F}_{ABB}$

Parameters: A modulus q that is either a prime or a power of 2.

Input:

- Upon input $(\mathsf{Input}, P_i, \mathsf{type}, \mathsf{id}, x)$ from P_i and input $(\mathsf{Input}, P_i, \mathsf{type}, \mathsf{id})$ from all other parties, if id is a fresh identifier and $(\mathsf{type}, x) \in \{\{\mathsf{binary}\} \times \mathbb{Z}_2, \{\mathsf{arithmetic}\} \times \mathbb{Z}_q)\}$ then store $(\mathsf{type}, \mathsf{id}, x)$.

Linear combination:

- Upon input $(\mathsf{LinComb}, \mathsf{type}, \mathsf{id}, (\mathsf{id}_j)_{j=1}^m, c, (c_j)_{j=1}^m)$ from all parties, if
  - each id_j is stored in memory, and,
  - $c, c_j \in \mathbb{Z}_2$ if type is binary and $c, c_j \in \mathbb{Z}_q$ if type is arithmetic,

  then
  - retrieve $((\mathsf{type}, \mathsf{id}_1, x_1), \ldots, (\mathsf{type}, \mathsf{id}_m, x_m))$,
  - compute $y \leftarrow \sum c_j \cdot x_j \mod 2$ if type is binary and $y \leftarrow \sum c_j \cdot x_j \mod q$ if type is arithmetic,
  - store $(\mathsf{type}, \mathsf{id}, y)$.

Multiplication:

- Upon input $(\mathsf{Mult}, \mathsf{type}, \mathsf{id}, \mathsf{id}_1, \mathsf{id}_2)$ from all parties, if $\mathsf{id}_1, \mathsf{id}_1$ are stored in memory then
  - retrieve $(\mathsf{type}, \mathsf{id}_1, x_1), (\mathsf{type}, \mathsf{id}_2, x_2))$,
  - compute $y \leftarrow x_1 \cdot x_2 \mod 2$ if type is binary and $y \leftarrow x_1 \cdot x_2 \mod q$ if type is arithmetic,
  - store $(\mathsf{type}, \mathsf{id}, y)$.

Converting from binary to arithmetic:

- Upon input $(\mathsf{ConvertB2A}, \mathsf{id}, \mathsf{id}')$ from all parties, if id' is present in memory then retrieve $(\mathsf{binary}, \mathsf{id}', x)$ and store $(\mathsf{arithmetic}, \mathsf{id}, x)$.

Output:

- Upon input $(\mathsf{Output}, \mathsf{type}, \mathsf{id})$ from all honest parties, if id' is present in memory then retrieve $(\mathsf{type}, \mathsf{id}, x)$ and output it to the adversary. Wait for input from the adversary of the form $(\mathsf{Deliver}, b)$, where $b \in \mathbb{Z}_2$. if $b = 1$ then output x to all parties, otherwise output $\perp$.

Figure 5: The ideal functionality for the arithmetic black-box.

C On SFE with DP Leakage

As noted shortly in Section 4, one cryptographic task that SIM*-CDP can handle but SIM+-CDP cannot is that of computing a differentially private mechanism whilst allowing the adversary to receive leakage throughout the protocol, as long as that leakage is DP, in particular when some leakage occurs before the corrupted party chooses their input. Joint computation of functions whilst allowing DP leakage has been studied in a few different settings with regards to output functions and adversarial models [5, 42, 59, 67]. Of particular interest to us is the work of [42] where it is proposed an ideal functionality in UC for this setting which is then realised with respect to private set intersection (PSI) in the presence of active corruptions. One reason that the ideal functionality of [42] cannot be expressed as an instantiation of SFE (the functionality used in SIM+-CDP) is that the functionality in [42] relaxes the guarantee of *input independence*, meaning that the corrupted party can choose their input based on the input of the honest party.

The PSI protocol of [42] outputs the exact set intersection (to only one of the parties, the other gets no output) and therefore their protocol as a whole intuitively cannot be SIM*-CDP. If one would instead realise their ideal functionality for computing a function with leakage, and enforce that all possible combinations of leakage functions and the output function to the corrupted party is DP (when seen as a composition), then SIM*-CDP can be achieved. Below in Figure 6 we re-iterate the ideal functionality from [42] but augmented to have two potentially different classes of leakage functions for each party. The need for this is that since f_1 and f_2 need not be the same, as in the case when only one of them gets an output, then one can allow the party whose output function is DP with better parameters to have leakage functions that use up more of the privacy budget. In Definition C.1 we reiterate [42]'s definition of SFE with DP leakage, reformulated for consistency with our notation.

Definition C.1 (SFE with DP leakage [42]). A protocol π securely realises f with leakage $(\mathcal{L}_1, \mathcal{L}_2)$ if π is a UC-secure protocol for $\mathcal{F}_{\text{SFE with leakage}}^{f, \mathcal{L}_1, \mathcal{L}_2}$ (see Figure 6). We say that the protocol realises f with $(\varepsilon_\kappa, \varepsilon_\kappa)$-SDP leakage if it realises f with $(\mathcal{L}_1, \mathcal{L}_2)$ if for every $(L_{ji}^{pre}, L_{ji}^{post}) \in \mathcal{L}_1 \cup \mathcal{L}_2$, the probabilistic function $D := D_1 || D_2 \rightarrow (L_{ji}^{pre}(D), L_{ji}^{post}(D))$ is $(\varepsilon_\kappa, \varepsilon_\kappa)$-SDP.

D Proofs

D.1 Proof of Proposition 4.2

Proof overview. We now prove Proposition 4.2, which in short says that in the plain model (without setup assumptions) there exists tasks and parameter regimes for which SIM+-CDP can be satisfied but not SIM*-CDP with the same parameters. This follows from the results that some ideal functionalities cannot be realised with UC security in the plain model, and this particularly holds for a large class of same-output probabilistic two-party functionalities, as proven in [11]. Such results yield the desired separation after noting that some optimal SDP mechanisms fall within that class of functionalities, and that they are directly computable with SIM+-CDP by use of general-purpose *standalone secure* two-party computation, which does not require setup assumptions.

Background. The UC part of the proof is in the *plain model*, meaning that no setup assumptions (such as having a common reference string) are made, and that one by default only has access to authenticated (not necessarily secure) channels [11, 12]. In the plain model, it has been shown that there are many functionalities that cannot be realised with UC-security. At the same time, it is known that

Functionality $\mathcal{F}^{f,\mathcal{L}_1,\mathcal{L}_2}_{\text{SFE with leakage}}$

Parameters:

- A function $f = (f_1, f_2) : (\{0,1\}^*)^2 \to (\{0,1\}^*)^2$.
- Two classes of functions $\mathcal{L}_j = \{(L^{pre}, L^{post})_{j1}, \ldots\}$, $j \in \{1,2\}$, with $L^{pre}_{ji}, L^{post}_{ji} : \{0,1\}^* \to \{0,1\}^*$.

No corruptions:

- Upon D_1 from P_1 and D_2 from P_2, deliver $f_1(D_1||D_2)$ to P_1 and $f_2(D_1||D_2)$ to P_2.

Party P_c corrupted (P_h is honest):

- Upon D_h from P_h and (Leak, L^{pre}) from P_c, if there exists an element $(L^{pre}, \cdot)$ in $\mathcal{L}_c$ then send $L^{pre}(D_h)$ to P_c, otherwise send $\perp$.
- Upon D_c and (Leak, L^{post}) from P_c, if there exists an element (L^{pre}, L^{post}) in $\mathcal{L}_c$ then send $L^{post}(D_h)$ to P_c, otherwise send $\perp$. Regardless, also send $f_c(D_1||D_2)$ to P_c.
- Upon (Deliver, b) from P_c, if $b = 1$ then send $f_h(D_1||D_2)$ to P_h, otherwise send $\perp$.

Figure 6: The ideal functionality for reactive two-party SFE with abort and leakage.

in the standalone model (also here without any setup assumptions apart from that there are authenticated channels), all two-party PPT functionalities can be realised. Further, any protocol that securely realises an ideal functionality computing an SDP mechanism in the standalone model also satisfies SIM$^+$-CDP with unchanged parameters. Therefore, if there exists a task for which there is an optimal mechanism and this mechanism can be realised with standalone security but not UC-security (without setup assumptions), then our desired protocol will follow. We now state some definitions and results needed for our proof.

Definition D.1 (Unpredictable function family [11]). A probabilistic function family $f = \{f_\kappa\}_{\kappa\in\mathbb{N}}$ with $f\kappa : \mathcal{D}^2 \to \mathcal{R}$ is said to be unpredictable if there exists a polynomial $p(\cdot)$ such that for infinitely many κ: $\exists D_1, D_2 \in \mathcal{D}$ such that:

(1) $\forall D_2' \in \mathcal{D}, z \in \mathcal{R} : \mathbb{P}(f_\kappa(D_1, D_2') = z) \leq \frac{1}{p(\kappa)}$.
(2) $\forall D_1' \in \mathcal{D}, z \in \mathcal{R} : \mathbb{P}(f_\kappa(D_1', D_2) = z) \leq \frac{1}{p(\kappa)}$.

Intuitively, f is unpredictable if (asymptotically) there are no choices of inputs any one of the parties can make such that the function output is almost always the same, regardless of the inputs of the other party. In other words, each party can choose its input such that the function output will not have almost all its probability mass at one output event. This is a pretty weak requirement on a probabilistic function. In particular, we have that the randomised response mechanism for binary functionalities (i.e. where for a binary function $f : \{0,1\}^2 \to \{0,1\}$, $f(D_1, D_2)$ is output with probability $\frac{e^\varepsilon}{e^\varepsilon+1}$ and otherwise its negation is output) is unpredictable, with p in the definition of unpredictability being chosen as a suitable constant. Further, it is easy to verify that for all binary functions, no pure SDP mechanism can have a higher probability of returning the true value than randomised response. Thus we can set u to be 1 iff the mechanism outputs the correct evaluation of a fixed arbitrary binary function, and α to be $\frac{e^\varepsilon}{e^\varepsilon+1}$, i.e. the probability of a correct answer when using randomised response with parameter ε.

Lemma D.2 (Reformulation of Theorem 6.1 in [11]). *Let $\mathcal{M} = \{\mathcal{M}_\kappa\}$ be a family of unpredictable PPT two-input same-output functions and let $\mathcal{F}$ be the ideal functionality that returns (to both players) a sample from $\mathcal{M}(D_1, D_2)$ when given D_1 from party 1 and D_2 from party 2. Then $\mathcal{F}$ cannot be UC-realised in the plain model by any non-trivial protocol.*

The notion of a non-trivial protocol is an extremely broad one, essentially only requiring that all parties get outputs in the case that all parties are honest and the adversary does not prevent any messages from being delivered. It is immediately clear that the protocol which realises the ideal functionality of randomised response in the standalone model with perfect correctness by use of general-purpose two-party computation is indeed non-trivial.

Proof of Proposition 4.2. What needs to be presented is a choice of ε and a task (α, u) together with proofs that it can be solved with $(\varepsilon, 0)$-SIM$^+$-CDP but not $(\varepsilon, 0)$-SIM*-CDP in the plain model. As explained in the preceding paragraphs, we define the utility function by choosing the boolean XOR function $f : \{0,1\}^2 \to \{0,1\}$ and set $u(D, z) = 1 \iff z = f(D_1, D_2) := D_1 \oplus D_2$. We set $\alpha := \frac{e^\varepsilon}{e^\varepsilon+1}$. Further, set ε such that there exists a polynomial p in κ such that α is a multiple of $2^{-p(\kappa)}$ for large enough κ. The randomised response mechanism with parameter ε, by construction, has α-usefulness for u, since α is exactly the probability with which the mechanism outputs the true answer. We may now make the following observations:

- There is no $(\varepsilon, 0)$-SDP mechanism with higher utility than α for u. This follows directly from a simple contradiction argument, namely that if a boolean mechanism has utility above α, then it cannot be $(\varepsilon, 0)$-SDP since there exists a choice of database such that the privacy loss random variable is above e^ε.
- If there exists a polynomial p in κ such that α is a multiple of $2^{-p(\kappa)}$ for large enough κ, then randomised response is computable in strict polynomial time, which implies (by the possibility of computational perfect-correctness general-purpose two-party computation in the standalone model, see e.g. Theorem 7.1.2 of [37]) that the ideal functionality that computes randomised response with respect to f can be realised in the standalone model.

The second observation above implies that there is a protocol which is $(\varepsilon, 0)$-SIM$^+$-CDP for the task (α, u), namely the one that uses general-purpose two-party computation to realise (with perfect correctness) the ideal functionality which performs randomised response with respect to f. The first observation above gives that randomised response is optimal in the sense that no other mechanism has higher utility and that any mechanism with utility indistinguishable from that of randomised response, also must have an output distribution which is indistinguishable from that of randomised response, due to the boolean output range.

Thus the only thing that remains to be shown is that there is no protocol that has a utility indistinguishable from that of the protocol above whilst satisfying $(\varepsilon, 0)$-SIM*-CDP. This follows directly from the fact that the randomised response ideal functionality is unpredictable (in the sense of Definition D.1) together with Lemma D.2. In particular, any $(\varepsilon, 0)$-SIM*-CDP protocol which has

utility indistinguishable from α has an output distribution that is indistinguishable from that the randomised response ideal functionality, which implies that the protocol UC-realises said functionality, which is a contradiction to Lemma D.2 since randomised response is unpredictable. Finally, we note that since the impossibility holds not only for protocols with exactly the utility α but also those with utility computationally indistinguishable from α, the impossibility result holds for SIM*-CDP protocols both with respect to real-world accuracy and ideal-world accuracy. □

D.2 Proof of Proposition 4.3

Proof overview. We now prove Proposition 4.3, which says that there are tasks such that in some parameter regimes, they can be solved with SIM*-CDP but not SIM$^+$-CDP. The whole idea of the proof is that, under sufficient complexity assumptions, both UC-security and standalone security allow general-purpose two-party computation, meaning that any PPT functionality can be realised. Now the requirement of perfect correctness in SIM$^+$-CDP means on the other side that no non-PPT ideal functionalities can be used to achieve SIM$^+$-CDP, whereas this is not the case for SIM*-CDP, since there the correctness requirement is computational rather than perfect. Again, we use the concrete proof strategy applied in Section D.1, namely showing that there is a parameter regime for which the randomised response mechanism can be realised with SIM*-CDP but not with SIM$^+$-CDP and this gives the result by the fact that randomised response is optimal for pure SDP boolean mechanisms.

Proof of Proposition 4.3. As in the proof of Proposition 4.2, define $u(D,z) = 1 \iff z = f(D_1,D_2) := D_1 \oplus D_2$ i.e. with f being the XOR function. Set $\alpha(\kappa) := \frac{e^{\varepsilon_\kappa}}{e^{\varepsilon_\kappa}+1}$. Now, as opposed to the previous proof, we set ε such that the resulting randomised response mechanism can *not* be computed in strict polynomial time. In particular, set ε_κ such that $\alpha(\kappa) = 1-2^{-2^\kappa}$, i.e. $\varepsilon_\kappa := \frac{2^{-2^\kappa}}{1-2^{-2^\kappa}}$. That is, the mechanism we consider is that in which $f(D)$ is returned except for which probability 2^{-2^κ}.

This mechanism can be realised (with computational correctness) with UC security under the common reference string (CRS) setup assumption, since that model allows general-purpose two-party computation (see, for instance [11, 12]). That is, with utility considered in the ideal world (i.e. the ideal functionality $\mathcal{F}$ is α-useful for u), the task (α, u) can be solved with SIM*-CDP.

On the other hand, the mechanism above can not be realised in the standalone model with perfect correctness, since it requires sampling a Bernoulli trial with parameter α, which is impossible in strict polynomial time since α is not a multiple of an inverse polynomial power of 2 (see Appendix A). Further, since randomised response is optimal for boolean functionalities, there is no $(\varepsilon_\kappa, 0)$-SDP mechanism that runs in strict polynomial time and has utility above α. Thus there is no PPT mechanism which is α-useful for u and consequently there is no $(\varepsilon_\kappa, 0)$-SIM$^+$-CDP protocol which is α-useful for u either. □

D.3 Proof of Proposition 4.4

We now prove Proposition 4.4, which in short says that for all protocols SIM*-CDP implies SIM-CDP with unchanged parameters. Note that this proof is analogous to that of the same relation between SIM$^+$-CDP and SIM-CDP, which is found in the long version of MPRV [64]. The proof follows essentially directly from the two definitions involved after noting that the mechanism (simulator) in SIM-CDP has access to all of the inputs and thus can run copies of the ideal world internally.

Proof of Proposition 4.4. Let $\varepsilon_\kappa \geq 0, \delta_\kappa \in [0,1]$ be arbitrary fixed classes of parameters. Let π be a two-party protocol that satisfies $(\varepsilon_\kappa, \delta_\kappa)$-SIM*-CDP with respect to some arbitrary fixed ideal functionality $\mathcal{F}$. This implies that for all PPT adversaries $\mathcal{A}$, the view of $\mathcal{S}_\mathcal{A}$ (the ideal-world adversary corresponding to $\mathcal{A}$) is $(\varepsilon_\kappa, \delta_\kappa)$-SDP and that the views of $\mathcal{A}$ and $\mathcal{S}_\mathcal{A}$ are computationally indistinguishable. That is, $\forall D \in \mathcal{D}, \mathcal{A} : \mathsf{VIEW}^{\mathcal{A}}_{\pi_{real}} \overset{c}{\approx} \mathsf{VIEW}^{\mathcal{S}_\mathcal{A}}_{\pi_{ideal}}$. This is due to the definition of UC-security, because if these two random variables are not computationally indistinguishable, then there exists an efficient environment that distinguishes the real and ideal worlds with a non-negligible advantage over guessing.

We can now turn the simulator $\mathcal{S}_\mathcal{A}$ into the mechanism $\mathcal{M}$ in the SIM-CDP definition by letting $\mathcal{M}$ run a copy of the ideal world protocol and then output the view of $\mathcal{S}_\mathcal{A}$. This is possible since in SIM-CDP, the mechanism (also at times called the simulator) $\mathcal{M}$ has access to the inputs of both the corrupted and honest parties. That is, since $\mathcal{M}$ can run a copy of the ideal world (thus making $\mathcal{M}(D)$ identically distributed to $\mathsf{VIEW}^{\mathcal{S}_\mathcal{A}}_{\pi_{ideal}}$), the indistinguishability of the views shown above implies that the view of the adversary in the real world is computationally indistinguishable from the output of $\mathcal{M}(D)$ for all D, which is $(\varepsilon_\kappa, \delta_\kappa)$-SDP, and thus π is $(\varepsilon_\kappa, \delta_\kappa)$-SIM-CDP. □

D.4 Proof of Lemma 5.4

Proof. Let $Z \sim \mathcal{M}^{p,f,\lambda}_{RTGeo}(D)$ and $Y \sim \mathcal{M}^{2B,f,\lambda}_{SRTGeo}(D)$ for arbitrary λ, D. Let p_Z and p_Y denote the probability density functions of Z and Y respectively and let F denote their cumulative distribution functions in the same manner. Since the parameter restrictions guarantee that the final sum in Y does not overflow (the result is as if the sum was done over the integers), the statistical distance between the two distributions is exactly twice the total probability mass that is affected by the truncation in Y. That is,

$$\begin{aligned}
SD(Z,Y) &= \frac{1}{2}\sum_{z\in\mathbb{Z}_p} |p_X(z) - p_Y(z)| \\
&= \sum_{z\in\mathbb{Z}_p\setminus(\bar{f}-B,\bar{f}+B)} |p_X(z) - p_Y(z)| \\
&= |F_X(\bar{f}-B) + (1 - F_X(\bar{f}+B))| \\
&= \left| \frac{e^{1/\lambda}}{e^{1/\lambda}+1} e^{-(\bar{f}-\bar{f}+B)/\lambda} \right. \\
&\quad \left. + \frac{1}{e^{1/\lambda}+1} e^{-(\bar{f}+B-\bar{f})/\lambda} \right| \\
&= e^{-B/\lambda},
\end{aligned}$$

where $\bar{f}$ is shorthand for $f(D)$. The equalities follow by inserting the formulas from Definition 5.1 and direct simplifications. □

D.5 Proof of Lemma 5.7

Proof. Firstly, $\mathtt{Ber}_{\hat{\alpha}}$ exactly samples a Bernoulli trial with a parameter equal to the recomposition of the first d elements of α. Call this parameter value α'. This means that the statistical distance between $\mathtt{Ber}(\hat{\alpha})$ and an exact Bernoulli trial with parameter $\hat{\alpha}$ is the same as between two exact Bernoulli trials with parameter $\hat{\alpha}$ and α', respectively. This statistical distance is equal to $|\hat{\alpha} - \alpha'|$, which is at most 2^{-d} since the first 2^d bits of their decomposition are identical.

Secondly, the statistical distance between $\mathcal{M}_{\mathsf{FDL}}^{\lambda,B,d,h}(D)$ and $\mathcal{M}_{SRTGeo}^{2B,h,\lambda}(D)$ is at most equal to the probability of any of the Bernoulli trials being incorrect, which due to independence is at most $B2^{-d}$. □

D.6 Proof of Lemma 5.8

Proof. The additive usefulness follows from a standard tail bound on the geometric distribution, since the truncated geometric is at least as concentrated as the untruncated one:

$$\begin{aligned}\mathbb{P}(|Geo_{q,\lambda}(f(D)) - f(D)| \geq \nu) &= \mathbb{P}(|Geo_{q,\lambda}(0)| \geq \nu)\\ &\leq \mathbb{P}(|Geo_{\lambda}(0)| \geq \nu)\\ &= 2F_{Geo_{\lambda}(0)}(-\nu)\\ &= \frac{2e^{1/\lambda}}{e^{1/\lambda}+1}e^{-\nu/\lambda}.\end{aligned}$$

□

D.7 Proof of Theorem 6.1

Proof. The definition of SIM*-CDP demands two things to be shown, namely that the view of the simulator is SDP and that the protocol UC-realises the ideal functionality. The first requirement is fulfilled as the only message sent from $\mathcal{F}_{SFE}^{\hat{f}_\kappa}$ to the corrupted party is $\mathcal{M}_{RTGeo}^{q,f_\kappa,\lambda_\kappa}(D)$ and this is $(\varepsilon_\kappa, 0)$-SDP due to the fact that the range-truncated geometric mechanism is $(\varepsilon_\kappa, 0)$-SDP under the standard parametrisation specified in the theorem. The other parts of the view of $\mathcal{S}$ (like its input and randomness tape) are independent of the inputs of the honest party, thus making the view of $\mathcal{S}$ as a whole $(\varepsilon_\kappa, 0)$-SDP. Further, this holds for all types of malicious behavour of $\mathcal{S}$ since, due to the formulation of $\mathcal{F}_{SFE}$, the only way $\mathcal{S}$ can change its view is to refuse to collaborate in the protocol or change its inputs and both of those decisions would have to be made independently of the inputs of the honest party (thus making those decisions $(0, 0)$-SDP as well).

The UC-realisation of the ideal functionality follows directly from the use of the arithmetic black-box and the statistical indistinguishability between $\mathcal{M}_{\mathsf{FDL}}$ and $\mathcal{M}_{RTGeo}$, which follows from lemmas 5.4 and 5.7 together with the assumptions of the theorem. In particular, due to the use of $\mathcal{F}_{ABB}$, the view of the corrupted party in the hybrid world consists of only its input, random coins and the output returned from $\mathcal{F}_{ABB}$, which is exactly $\mathcal{M}_{\mathsf{FDL}}$. Similarly, the view of the corrupted party in the ideal world is also only its input, random coins and output returned from $\mathcal{F}_{SFE}^{\hat{f}_\kappa}$. Therefore the simulator that simply outputs its view (after having changed its inputs and/or aborted with respect to its random coins as the hybrid-world adversary does) yields a view that is computationally indistinguishable from that of the hybrid-world adversary. Further, this simulator is strict PPT due to it performing only the same operations as the hybrid-world adversary (choosing input and abort behaviour based on its coins and then receiving one $\mathbb{Z}_q$ element), hence the theorem follows. □

E Techniques for Achieving Secure MPC

In the context of MPC, we typically distinguish binary and arithmetic protocols. This classification describes the possible computations. In other words, we perform addition and multiplication in $\mathbb{F}_2$ and $\mathbb{F}_p$, respectively. In this work, we rely on secret sharing-based (SS) MPC protocols. More precisely, we use additive secret sharing (ASS). In such protocols, secret values x are shared among n parties by uniformly sampling $n-1$ random values $x_1, \ldots, x_{n-1}$ from $\mathbb{F}$, setting $x_0 \leftarrow x - \sum_{i=1}^{n} x_i$, and distributing x_i to every party p_i. We denote secret shared values as $[\![x]\!]$. We further denote $[\![x]\!] \leftarrow \mathsf{Share}(x)$, and $x \leftarrow \mathsf{Reconstruct}([\![x]\!])$ as sharing and reconstructing secrets. ASS schemes are additively homomorphic, allowing the addition of shares without interaction and hiding underlying secrets as long as there is one honest party. To allow multiplications with an ASS, one can use multiplication triples, introduced by Beaver [3]. Triples are three shared values $([\![a]\!], [\![b]\!], [\![c]\!])$, that no party knows and that fulfil $a \cdot b = c$. When multiplying two shared values $([\![x]\!], [\![y]\!])$, one reconstructs masked versions $\alpha \leftarrow \mathsf{Reconstruct}([\![x]\!] - [\![a]\!])$, $\beta \leftarrow \mathsf{Reconstruct}([\![y]\!] - [\![b]\!])$, and computes[22] $[\![z]\!] \leftarrow \alpha\beta + \beta[\![x]\!] + \alpha[\![y]\!] + [\![c]\!] = [\![x \cdot y]\!]$.

Given these ingredients, we can instantiate an active secure general-purpose MPC protocol if we have access to a secure sampling method for multiplication triples, and adversaries cannot tamper with the reconstruction procedure. In the SPDZ paper [23], the authors introduced solutions to both problems. They propose an additively homomorphic encryption scheme for sampling triples and information-theoretic message authentication codes (MACs) to secure the reconstruction procedure. Subsequent work introduced several performance improvements by instantiating the ASS over the ring $\mathbb{F}_{2^k}$ [20] or replacing the expensive homomorphic encryption with oblivious transfer [52]. Note that both improvements, to some degree, accept a higher communication for a lower computation complexity.

[22] This step requires multiplication and addition with constant terms which follows from the ASS properties.