Additive Randomized Encodings and Their Applications

Shai Halevi[*] Yuval Ishai[†] Eyal Kushilevitz[†] Tal Rabin[‡]

June 7, 2023

Abstract

Addition of n inputs is often the easiest nontrivial function to compute securely. Motivated by several open questions, we ask what can be computed securely given only an oracle that computes the sum. Namely, what functions can be computed in a model where parties can only encode their input locally, then sum up the encodings over some Abelian group $\mathbb{G}$, and decode the result to get the function output.

An *additive randomized encoding* (ARE) of a function $f(x_1, \ldots, x_n)$ maps every input x_i independently into a randomized encoding $\hat{x}_i$, such that $\sum_{i=1}^{n} \hat{x}_i$ reveals $f(x_1, \ldots, x_n)$ and nothing else about the inputs. In a *robust* ARE, the sum of *any subset* of the $\hat{x}_i$ only reveals the *residual function* obtained by restricting the corresponding inputs.

We obtain positive and negative results on ARE. In particular:

- *Information-theoretic ARE.* We fully characterize the 2-party functions $f : X_1 \times X_2 \to \{0, 1\}$ admitting a perfectly secure ARE. For $n \geq 3$ parties, we show a useful "capped sum" function that separates statistical security from perfect security.
- *Computational ARE.* We present a general feasibility result, showing that *all functions* can be computed in this model, under a standard hardness assumption in bilinear groups. We also describe a heuristic lattice-based construction.
- *Robust ARE.* We present a similar feasibility result for *robust* computational ARE based on ideal obfuscation along with standard cryptographic assumptions.

We then describe several applications of ARE and the above results.

- Under a standard cryptographic assumption, our computational ARE schemes imply the feasibility of general non-interactive secure computation in the *shuffle model*, where messages from different parties are shuffled. This implies a general utility-preserving compiler from differential privacy in the central model to computational differential privacy in the (non-robust) shuffle model.
- The existence of information-theoretic *robust* ARE implies "best-possible" information-theoretic MPC protocols (Halevi et al., TCC 2018) and degree-2 multiparty randomized encodings (Applebaum et al., TCC 2018). This yields new positive results for specific functions in the former model, as well as a simple unifying barrier for obtaining negative results in both models.

[*]Algorand Foundation
[†]Technion - Israel Institute of Technology
[‡]University of Pennsylvania

Contents

1 Introduction

Secure multiparty computation (MPC) [Yao86, GMW19, BOGW88, CCD88] enables n parties to evaluate a distributed function $f(x_1, \ldots, x_n)$ on their local inputs, while revealing nothing except the output of f. Most of the questions about the general *feasibility* of MPC in different models have already been settled, shifting the focus of most research in the area to improved *efficiency*. The current work is motivated by several remaining questions on the feasibility front. Unless stated otherwise, we consider here security in the presence of *semi-honest* parties, who send messages as instructed by the protocol.

- **Non-interactive MPC in the shuffle model [IKOS06].** Is it possible to compute every function f securely (with either information-theoretic or computational security) by having the parties simultaneously send *anonymous* messages to an evaluator? The evaluator should be able to recover $f(x_1, \ldots, x_n)$ from the shuffled messages, but learn nothing else about the inputs.
- **Best-possible information-theoretic MPC [HIKR18].** Does every function f admit an information-theoretic MPC protocol that offers security against $t < n/2$ corrupted parties while only revealing to $t \geq n/2$ corrupted parties the *residual function* of f obtained by fixing the inputs of honest parties?
- **Minimal complete primitive for MPC [ABT21, ABG$^+$20].** Is it possible to compute every function f with information-theoretic security against any number of corrupted parties, by using a single call to a degree-2 function g, or alternatively parallel calls to functions $g_{i,j}$ that depend on only 2 inputs and make their outputs public?

When f is just the n-party addition function over a finite Abelian group, then the answer to all of the above questions is "yes." The addition function is also attractive from an (asymptotic and concrete) *efficiency* perspective, and several lines of works propose optimized protocols and applications of secure addition in the context of federated learning [BIK$^+$17], private analytics [CB17], and more. In light of this, it is natural to ask:

> What can be computed securely given only an oracle for addition?

Additive Randomized Encoding. We capture this question via the new notion of an *additive randomized encoding* (ARE). Given a function $f(x_1, \ldots, x_n)$ and an Abelian group $\mathbb{G}$, an ARE scheme for f over $\mathbb{G}$ is defined by n randomized local encoding functions Enc_i, mapping each input x_i to a group element $\hat{x}_i \in \mathbb{G}$, and a decoder $\mathsf{Dec}(\hat{y})$, mapping the sum of the encodings $\hat{y} = \sum_{i=1}^{n} \hat{x}_i$ (over $\mathbb{G}$) to an output y. It will sometimes be convenient to consider ARE as a non-interactive *protocol*, referring to $\hat{x}_i$ as the ARE message of party P_i.

An ARE as above should satisfy the following correctness and security requirements. The *correctness* requirement is that the above process results in $y = f(x_1, \ldots, x_n)$. The *security* requirement is that the sum $\hat{y}$ reveals nothing about the inputs except the output of f. This can be viewed as an instance of the standard notion of a randomized encoding (RE) of functions [AIK06], where the encoding $\hat{f}$ of f is restricted to "adding up local randomized functions." See Section 1.3 for further discussion of the relation with RE and its multiparty variant from [ABT21].

As a simple example, consider the following ARE for the OR of n input bits: given a group $\mathbb{G}$, an input $x_i = 0$ is encoded as $\hat{x}_i = 0$ and $x_i = 1$ is encoded as a uniformly random element of $\mathbb{G}$. If $\mathrm{OR}(\mathbf{x}) = 0$ then the sum $\hat{y}$ of encodings is 0, while if $\mathrm{OR}(\mathbf{x}) = 1$ then $\hat{y}$ is random in $\mathbb{G}$, so security is perfect. Correctness error occurs with probability $1/|\mathbb{G}|$, if $\mathrm{OR}(\mathbf{x}) = 1$ but the sum of the ARE messages turns out to be 0 by chance.

Robust ARE. The above notion of ARE is natural when the encoded output $\hat{y}$ is revealed to an external party, who does not collude with any of the n parties. For the case where $\hat{y}$ can be learned by a coalition $T \subseteq [n]$ of corrupted parties, we need to account for the fact that corrupted parties can learn the sum $\hat{y}_H$ of the encodings $\hat{x}_i$ generated by the set of honest parties $H = [n] \setminus T$. This allows them to compute the value $f(x_T^*, x_H)$ for any x_T^* of their choice. We say that an ARE is *robust* if this is the only information that can be deduced from $\hat{y}_H$. It is not hard to verify that the above ARE scheme for OR is robust in this sense.

1.1 Our Contribution

In this work we study the feasibility of ARE and robust ARE for general functions f, with perfect, statistical, and computational security, apply our positive results towards solutions for the first two motivating questions discussed above, and highlight ARE as a new barrier for obtaining negative answers to the last two motivating questions.

Information-theoretic ARE. In the information-theoretic setting, we obtain both positive and negative results. In particular, we fully characterize the 2-party functions $f : X_1 \times X_2 \to \{0, 1\}$ admitting a *perfectly secure* (but possibly statistically correct) ARE. These are precisely the functions f that can be written as $g(f_1(x_1), f_2(x_1))$ for some Boolean functions f_1 and f_2. This means that, when insisting on perfect security, OR and XOR are the only two-party Boolean functions that can be realized (up to local preprocessing of inputs and postprocessing of the output).

For $n \geq 3$ parties, we consider a natural "capped sum" function that adds up integer-valued inputs and reveals the sum only if it is at most (alternatively, at least) some predetermined threshold θ. (Otherwise the only bit of information revealed is that the threshold has been met.) We also consider a variant that includes "payloads," that are revealed when the sum does not exceed the threshold. The capped sum functionality is motivated by applications related to anonymous communication. We present an ARE for capped sum (including the "payload" variant) in which both correctness and security are statistical, and apply this towards an ARE for multiplication modulo an arbitrary integer m. We prove that there is no perfectly secure ARE for capped sum, thus providing a provable separation between perfect and statistical security for ARE. All of our constructions of information-theoretic ARE schemes are in fact robust. We leave open the existence of *statistically secure* ARE for general (or even constant-size) functions, and describe a failed attempt in this direction in Appendix A.

Computational ARE. We present a general feasibility result, showing that *all* polynomial-time computable functions admit a computationally secure ARE, under a standard hardness assumption in bilinear groups. An optimized variant of this scheme is quite practical: each party needs to send only a constant number of group elements per input bit, and additionally one of the parties needs to generate and send a standard garbled circuit for f. We also describe a heuristic lattice-based construction that we conjecture to be secure. These constructions do *not* realize the stronger notion of *robust* ARE that we discuss next.

Robust computational ARE. The necessity of revealing the *residual* function in robust ARE means that robust ARE for general functions (over large input domains) implies obfuscation. We show that the converse is in a sense also true. Using *resettable* MPC [GS09, GM11] (which can be based on standard cryptographic assumptions), we obtain a general feasibility result for robust ARE based on *ideal obfuscation*.[1] The ideal obfuscation model, which was used to establish several recent results in cryptography, is similar in spirit to other generic models, such as the random oracle model. A formal support for this view was recently given in [JLLW22]. To get a (heuristic) standard-model construction, the ideal obfuscation oracle can instantiated using indistinguishability obfuscation. Alternatively, our robust ARE can be provably realized (under standard assumptions) in the *pseudorandom oracle model* of [JLLW22].

Application 1: Non-interactive MPC in the shuffle model. The *shuffle model* assumes that parties can send anonymous messages which are effectively shuffled before arriving to their destination. (Each party may send multiple messages.) First studied in the context of MPC [IKOS06], the shuffle model is currently a popular model for distributed *differential privacy* [CSU$^+$19, EFM$^+$20], offering better privacy-utility tradeoffs than the distributed *local model* while requiring less trust than the *central model*.

An MPC protocols for *addition* in the shuffle model was given in [IKOS06] (see [GMPV20, BBGN20] for a tighter analysis). Combining this protocol with our computational ARE constructions, we get the first general feasibility results for (computationally secure, non-interactive) MPC in the shuffle model. Depending on the kind of ARE used, one either gets a non-robust (and practical) protocol under a standard cryptographic assumption, or an optimally robust (but currently impractical) protocol under strong assumptions. The ARE-based protocols do not involve any setup, thus providing qualitative advantages over alternative models for non-interactive MPC such as the PSM model [FKN94] or robust non-interactive MPC with public-key setup [HIJ$^+$17]. The non-robust variant of the protocol implies a general utility-preserving compiler from differential privacy in the central model to computational differential privacy in the (non-robust) shuffle model. We do not know how to usefully apply robust ARE towards differential privacy in the robust shuffle model, and leave this as an interesting open question.

Application 2: Best-possible Information-Theoretic MPC. The notion of Best-possible Information-Theoretic MPC (BIT-MPC), introduced in [HIKR18], considers a stronger variant of the standard notion of information-theoretic security for MPC, which is in a sense the best possible. In the standard notion of IT-MPC for f, there are $t < n/2$ corrupted parties, and the protocol should guarantee that corrupted parties learn nothing except the output. The notion of BIT-MPC makes the additional requirement that even a majority of t corrupted parties should not learn more than the *residual function* obtained from f by fixing the inputs of the honest parties. It was shown in [HIKR18] that this information must be leaked, hence BIT-MPC indeed provides the best possible security. The main question left open by [HIKR18] is whether *all* functions can be realized in this

[1] A combination of resettable MPC and ideal obfuscation was informally proposed in [HIJ$^+$17] in the related context of non-interactive MPC (see Section 1.3). It was recently used in [BIK$^+$22] in a very different context: constructing a counterexample to a dream version of Yao's XOR lemma.

model, regardless of efficiency. We connect this question to ARE, by showing that any perfectly (resp., statistically) secure robust ARE for f implies a perfectly (resp., statistically) BIT-MPC protocol for f. This allows transferring our positive results, such as the ones for variants of capped-sum, into the BIT-MPC setting. Moreover, obtaining negative results for (robust) statistical ARE is a necessary condition, which may be viewed as a *barrier*, for ruling out BIT-MPC protocols for general functions.

Application 3: Barrier for Multiparty Randomized Encoding. Finally, we show that a (hypothetical) robust information-theoretic ARE would imply an optimal construction of *multiparty randomized encodings* (MPRE) [ABT21] and thus, again, can be viewed as a barrier for settling a well-known open problem. In an MPRE for $f(x_1, \ldots, x_n)$, each input x_i can be preprocessed "for free" by a local encoder before feeding it into a global encoder $\hat{f}$ whose output is made public. MPRE requires that even from the point of view of *insiders*, the output of $\hat{f}$ must reveal no more information about the other inputs than what follows from their own inputs and the output of f. The (effective) *degree* of an MPRE is the algebraic degree of $\hat{f}$. ARE can be viewed as a variant of MPRE with degree 1, but where security is only guaranteed against an outsider who observes the output. Robust ARE also falls short of meeting the MPRE requirement because of the leakage of the residual function. However, we show that robust ARE can still be used to construct standard MPRE with degree 2. This should be contrasted with the best known information-theoretic MPRE constructions, which either have degree 3 [ABT21, ABG$^+$20] or alternatively have degree 2 but are only secure against $2n/3$ corrupted parties [AIKP22]. In fact, our transformation only requires a robust MPRE for a simple (and constant-size) 3-party function. Thus, a robust statistical ARE for such functions would settle the main open question in this area.

We note that while our computational construction of robust ARE implies *computational* degree-2 MPRE, such a result was already known based on the much weaker assumption that oblivious transfer exists [ABT21]. The significance of the ARE-to-MPRE transformation is that it gives another barrier for ruling out degree-2 MPRE: such a negative result requires ruling out statistical robust ARE for a constant-size 3-party function. Another (23-year old) barrier for the MPRE question is ruling out standard statistically secure degree-2 RE for general functions [IK00, ABG$^+$20]. While the two barriers seem technically incomparable (see Section 1.3 below for discussion), we believe that the ARE barrier may be easier to make progress on because of the simpler additive structure. For the BIT-MPC question, ARE give the first simple barrier for proving negative results: the degree-2 RE barrier does not seem to apply, since BIT-MPC protocols are not known to follow from degree-2 RE.

1.2 Open Questions

Our work leaves many questions about ARE open. The main open question is whether *all* functions admit a statistically secure (robust or non-robust) ARE. This is open even for very simple functions, such as equality of two inputs from the domain $\{0, 1, 2\}$, which can be shown to be complete for non-robust ARE (Theorem 5.10). We strongly conjecture that the answer is negative. However, we were not able to prove this conjecture, and document in Appendix A the closest we got: a negative result under an alternative security definition that replaces the standard l_1 (statistical) distance between distributions by l_2 distance.

Other questions for information-theoretic ARE include extending our full characterization for *perfectly* secure ARE beyond two parties, and ruling out an ARE for OR with perfect correctness.

For (non-robust) computational ARE, the main question is to better understand the required assumptions. Can we use other "public-key" assumptions, such as DDH or LWE? Is public-key cryptography even needed? Are there any general connections with other cryptographic primitives?

Finally, can we construct *robust* ARE (for constant-size functions) from iO and standard cryptographic assumptions, avoiding the use of ideal obfuscation?

1.3 Related Work

Randomized encoding of functions. It is often useful to replace a given function f by a "simpler" randomized function $\hat{f}$ whose output can be used to recover the output of f but reveals no additional information about the input. This was formalized by the abstract notion of *randomized encoding* (RE) of functions [IK00, AIK06], which has found many applications in cryptography and beyond (see [Ish13, App17] for surveys). ARE can be viewed as an instance of RE where the notion of simplicity is "adding up local randomized functions."

The main open question about RE is the existence of *statistically secure* degree-2 RE for all (constant-size) functions. This question, first posed in [IK00], is open for over 20 years, and has been put forward as a barrier for solving other questions [AHI$^+$17, ABG$^+$20]. We note that the class of functions admitting (statistical) degree-2 RE seems incomparable to the class of functions admitting (statistical) ARE, even when restricting the ARE model to allow a single bit of input per party. On the one hand, the mod-2 inner product function trivially has a degree-2 RE whereas it is conjectured not to have ARE. (In the case of perfect security, this provably follows from our 2-party characterization.) On the other hand, the capped sum function has (statistical) ARE, but it seems reasonable to conjecture that it has no degree-2 RE. (A natural implementation of our ARE scheme in the RE setting would lead to a degree-3 RE.)

Multiparty randomized encoding. As discussed above, MPRE [ABT21] is a natural extension of RE to the multi-party setting. Viewed as an MPRE, ARE maximizes the simplicity of the global encoder, but (inherently) does so at the expense of sacrificing full security against insiders. For standard (non-robust) ARE, the only security requirement is against an outsider who obtains the output of $\hat{f}$, whereas a robust ARE offers the best-possible security against insiders.

Non-interactive MPC. ARE serves as a natural tool for non-interactive MPC in different models. Unlike existing models for non-interactive MPC, including the well-studied PSM model [FKN94] or its robust variants from [BGI$^+$14, HIJ$^+$17, AAP19], ARE does not require any form or correlated randomness or public-key setup.

Organization. In Section 2 we provide a technical overview of our results. A more detailed treatment can then be found in Sections 3 (definitions), 4 (information-theoretic constructions and lower-bounds), 5 (computational constructions), 6 (robust ARE), and 7 (some applications).

2 Overview of Techniques

In this section we give a detailed but informal overview of the technical ideas behind our main results.

2.1 Information-Theoretic ARE

Our positive results for information-theoretic AREs, consist of randomization techniques for the functions in question. We start by recalling the simple example of OR of n input bits described in the introduction: $x_i = 0$ is encoded as a 0, while $x_i = 1$ is encoded as a uniformly random value in $\mathbb{G}$. If $\mathrm{OR}(\mathbf{x}) = 0$ then the sum of encodings is 0 while if $\mathrm{OR}(\mathbf{x}) = 1$ then the sum of encodings is random in $\mathbb{G}$. Privacy is therefore perfect, and correctness error occurs with probability $1/|\mathbb{G}|$. Note that this is in fact a *robust* ARE, since the sum of a subset of the ARE messages only reveals the OR of the corresponding inputs, which coincides with the residual function restricted to these inputs. [2] This OR protocol can be extended to compute the MAX function (maximum of n integers), though the communication complexity in this case scales exponentially with the bit-length of the inputs.

A more interesting example is the *capped-sum* function where the output is the sum of the inputs (over the integers), unless the sum exceeds some pre-set cap θ, in which case the output is just θ. Beyond being a natural example, it also serves as a building block for other constructions (such as ARE for product modulo m) and can be extended to a variant with payloads that can be motivated by anonymity-related applications. To get an ARE scheme for capped sum, each input x_i is encoded as a random $\theta \times \theta$ matrix with rank x_i, over a sufficiently large finite field. The observation is that up to θ, the rank of the sum is equal, with high probability, to the sum of the ranks: in that range, a sum of random matrices of ranks x_i is close to a random matrix of rank $\sum_i x_i$. And, of course, the rank can never exceed θ. In this case, both correctness and security of the ARE are statistical.

We also present several negative results for perfectly-secure information-theoretic AREs. Among other things, we show that the above capped sum function cannot have a perfectly-secure ARE, and the statistical security in our solution is necessary. In this overview, it is convenient to restrict attention to two-argument functions $f(x, y)$. The (randomized) encoding for every value of x is associated with some probability distribution p_x and, similarly, every y is associated with some probability distribution q_y. The sum of the two encodings for inputs (x, y) is therefore distributed according to the *convolution* of the two distributions, $p_x * q_y$. A convenient way to look at convolutions is by switching to the Fourier representation of the distributions. While each entry in $p_x * q_y$ (viewed as vectors) depends on all the entries of p_x and q_y, in the Fourier representation each entry depends only on the corresponding entries in the transforms of p_x and of q_y. Namely, $\widehat{p_x * q_y} = \widehat{p}_x \odot \widehat{q}_y$, where $\odot$ denotes entry-wise product and $\widehat{p}$ denotes the Fourier representation of p.

We then define a notion of a *Vector Multiplication Program* (VMP) for the function f. This is a collection of (complex) vectors v_x for each input value x, vectors w_y for each input value y, and distinct vectors u_z for each output value z, such that for all (x, y) we have $v_x \odot w_y = u_{f(x,y)}$. It follows from the above discussion that if there is a perfectly-secure ARE for $f(x, y)$ then there is also a VMP for f. Using the simplicity of the VMP framework, we are able to obtain, for example, an exact characterization of the two-argument Boolean functions that admit a perfectly-secure ARE. Concretely, this is exactly the set of functions $f(x, y)$ that can be expressed as $g(f_1(x), f_2(y))$, for Boolean f_1, f_2.

[2] In the following we will not refer to robustness, though all of our ARE constructions in the information-theoretic setting are in fact robust.

2.2 Computational ARE

Central to our treatment of computational ARE is the observation that the *two-party equality function* is complete in some sense, even over domains of fixed size (see below). We therefore begin by describing a pairing-based two-party ARE scheme for the equality function. The starting point for that scheme is considering the equation

$$(s_1 + s_2)(x_1 s_1 - x_2 s_2) \stackrel{?}{=} (x_1 s_1^2 - x_2 s_2^2)$$

and noting that the cross-terms $s_1 s_2$ are canceled out if and only if $x_1 = x_2$. This suggests a protocol where $P_1(x_1)$ sends $(s_1, x_1 s_1, x_1 s_1^2)$ for a random s_1, and $P_2(x_2)$ sends $(s_2, -x_2 s_2, -x_2 s_2^2)$ for a random s_2. If $x_1 = x_2 = x$, then the sum of these two vectors is of the form $(s_1 + s_2, x(s_1 - s_2), x(s_1^2 - s_2^2))$, and the evaluator can check that the last element equals the product of the first two. If $x_1 \neq x_2$ then the product of the first two will have additional terms that depend on the random s_1, s_2, so it will not be equal to the third term (except with a negligible probability).

Of course, sending the terms above "in the clear" will be insecure, in particular the evaluator can learn the difference $x_1 - x_2$. To avoid that, we encode those terms "in the exponent" and rely on DDH-like assumptions. Since evaluation requires computing the product, we use pairing-friendly groups that allow us to perform this multiplication in the exponent. More details are provided in Section 5.1. In the appendix we also describe a heuristic construction that attempts to replicate this structure with a lattice-based construction.

Once we have a scheme for equality, we can build from it a scheme for all other boolean functions with small domains: Party $P_2(x_2)$ prepares a list of all the possible inputs x_1 such that $f(x_1, x_2) = 1$, then run an equality ARE for each one to see if P_1's input is any of them. To ensure security, P_2 needs to pad the list so as to always run the same number of equality tests, and shift it by a random amount to hide which of these instances matches. See Section 5.2.

Having access to ARE schemes for all boolean functions of small domains, we use it to get 1-out-of-2 oblivious-transfer (OT), and combine it with a standard garbling technique (e.g., [AIK06]) to get a computational ARE for any multiparty function f. That is, one party (say P_1) will prepare a garbled circuit and send it to the evaluator, and will also engage in an OT instance with each other party for each bit of the input labels for that circuit. See Section 5.3. We also observe that the structure of the equality ARE scheme from Section 5.1 makes it easy to modify so as to get directly a scheme for OT, bypassing some of the generic transformations above. This optimization, described in Section 5.1.3, leads to a practical protocol in which the communication includes a standard garbled circuit along with a constant number of group elements per input bit.

2.3 Robust ARE

We note that general robust ARE over large input domains implies obfuscation. We therefore construct robust ARE using obfuscation. A natural approach is to start from a "sufficiently robust" interactive protocol, and obfuscate its next-message function. Of course, doing so means that the adversary can reset the honest parties, so the underlying interactive protocol must achieve best-possible security even with a resetting adversary.

A first attempt at getting a robust ARE for $f(x_1, \ldots, x_n)$ is therefore to start from a resettable-secure MPC protocol for f [GS09, GM11], then obfuscate its next message function (with the input and randomness of each party hard-wired). Of course, this does not use the summation oracle of ARE, and so (unsurprisingly) it is insecure. To see the problem, note that the adversary can reset *any subset of the honest parties*, then run the protocol with these parties fixed and the inputs to all other parties chosen arbitrarily. In other words, the adversary gets access to the *full residual function* of $f(x_1, \ldots, x_n)$, where it can substitute *any subset* of the inputs. In contrast, a robust ARE scheme can only leak the (standard) residual function, where the inputs of *all the honest parties* are fixed.

To do better, we extend the function f in a way that allows us to "lock" the inputs of the honest parties. To wit, we consider the extended function

$$g_f\big((x_1, \rho_1, \sigma_1), \ldots, (x_n, \rho_n, \sigma_n)\big) = \begin{cases} f(x_1, \ldots, x_n) & \text{if } \sigma_1 = \cdots = \sigma_n = \bigoplus_{i \in [n]} \rho_i, \\ \perp & \text{otherwise.} \end{cases}$$

Similarly to previous applications combining ideal obfuscation and resettable MPC [HIJ+17, BIK+22], we use this mechanism to emulate an ideal access to the *full* residual function of g_f, enabling the adversary to fix the inputs of *any strict subset* of the parties. This is done by sending obfuscations of the next message functions of a resettable MPC protocol for g_f, where (similarly to [BIK+22]) the protocol works over broadcast channels and uses signatures to enforce in-order executions. In addition to the full residual function of g_f, the sum $\sigma = \rho_1 + \ldots + \rho_n$ is communicated to the evaluator via the ARE.

The key point is that given oracle access to the full residual function of g_f, the adversary cannot predict the sum of the ρ_i's of a strict subset of the honest parties. The only way for the adversary to get a matching σ is to use the one that the evaluator received. But this σ ties it also to the ρ_i's of all the other honest parties. Hence, if the adversary uses any inputs of the honest parties then it must use them all.

2.4 Applications

Application 1: MPC in the shuffle model. The shuffle model was discussed and motivated above. Here, we outline our construction of MPC protocols in this model. Such a protocol for a function f is based on two ingredients. The first is an ARE for f over some group $\mathbb{G}$. Using this ARE, our protocol starts by each party P_i locally encoding its input x_i into $\hat{x}_i$. Next, we would like to compute the encoded output $\hat{y}$ which is just $\sum_i \hat{x}_i$. This is done, using an addition protocol for the shuffle model from [IKOS06] (tightly analyzed in [GMPV20, BBGN20]). In this protocol, the messages of each party are an additive secret sharing (over $\mathbb{G}$) of its input. The value $\hat{y}$ is reconstructed by adding up the shuffled shares, and then the ARE decoder is applied to obtain $y = f(\mathbf{x})$.

Application 2: From ARE to Best-Possible Information-Theoretic MPC. The notion of Best-possible Information-Theoretic MPC (BIT-MPC), introduced in [HIKR18], considers protocols that provide best possible type of security, depending on the number of dishonest parties. Namely, it offers the standard notion of security against a corruption of a minority of parties and, additionally, offers residual security in case that the adversary corrupts a majority of the parties. The work of [HIKR18] provides BIT-MPC protocols for certain families of functions (such as OR, and deciding the solutions of a system of linear equations $Ax = b$) but rules out *efficient* BIT-MPC for all efficiently computable functions. They leave open the question of the possibility of non-efficient BIT-MPC for all functions, or BIT-MPC for all constant-size functions.

We show that information-theoretic robust ARE for a function f can be transformed into a BIT-MPC for f, hence enriching the set of functions for which such BIT-MPC protocols are known. Moreover, this also implies that proving negative results for BIT-MPC requires proving impossibility of statistical ARE, explaining our difficulty of proving such results.

Given such an ARE for f, we construct a BIT-MPC protocol for f as follows. Each party P_i first uses the ARE to (locally) encode its input x_i into $\hat{x}_i$. Then, the parties employ a simple n-secure addition protocol that computes an additive secret sharing of $\hat{y} = \sum_i \hat{x}_i$. Finally, they use a standard information-theoretic MPC protocol (such as the BGW protocol [BOGW88]), secure in the presence of honest majority, to compute the ARE decoder on the sum of additive shares, which results in the desired output $f(x_1, \ldots, x_n)$. To argue BIT security, note that if there is a honest majority, then nothing beyond the output is revealed. On the other hand, if there is a dishonest majority, then the adversary may learn $\hat{y}$. However, the robustness of the ARE, implies that no more than the residual function of the honest parties' inputs is leaked by $\hat{y}$.

Application 3: Barrier for Multiparty Randomized Encoding. To obtain degree-2 MPRE from robust ARE, our starting point is that every function g that can be written as the sum of 2-local functions $f_{ij}(x_i, x_j)$ admits a degree-2 MPRE. The main technical challenge is leveraging robust ARE towards constructing an MPRE of the above form for general functions.

A key difference between the robust ARE model and the MPRE model is that the former (inherently) has residual function leakage whereas the latter (by requirement) does not. To eliminate the residual function leakage, we convert $f(x_1, \ldots, x_n)$ into a new N-party function f', for $N = \binom{n}{2}$ "virtual parties," which applies a simple *constant-size* pairwise multiparty authentication for the inputs of f. Concretely, for each pair of parties $1 \leq i < j \leq n$, there is a virtual party P_{ij} whose input to f' is a pair $x^{i,j} = (x_i^{i,j}, x_j^{i,j})$. The function f' checks that all input pairs are consistent with some global input vector $(x_1, \ldots, x_n)$, outputting $f(x_1, \ldots, x_n)$ if it is and $\perp$ otherwise. Namely,

$$f'(x^{1,2}, \ldots, x^{n-1,n}) = \begin{cases} f(x_1, \ldots, x_n) & \text{if } \exists x_1, \ldots, x_n \text{ s.t. } \forall i, j, x^{i,j} = (x_i, x_j) \\ \perp & \text{otherwise} \end{cases}.$$

Note that if f has constant-size input domains then so does f'.

We now use a robust ARE for f' to define an MPRE for f in which the function g is a sum of 2-local functions, which (as noted above) suffices for our purposes. The function g takes from each party P_i the following inputs: its original input x_i, and additional inputs ρ_{ij} (for all $j \neq i$) that will be used to generate the ARE messages of virtual parties P_{ij}.

Letting $\Pi = (\mathsf{Enc}_{ij}, \mathsf{Dec})$ be a robust ARE for f' over a group $\mathbb{G}$, the function g is defined as:

$$g\left(\left(x_1, (\rho_{1j})_{j \neq 1}\right), \ldots, \left(x_n, (\rho_{nj})_{j \neq n}\right)\right) = \sum_{1 \leq i < j \leq n} \mathsf{Enc}_{ij}((x_i, x_j); \rho_{ij} \oplus \rho_{ji})$$

where summation is over $\mathbb{G}$, and $\mathsf{Enc}_{ij}((x_i, x_j); \rho)$ denotes an ARE encoding for f' of input (x_i, x_j) (for virtual party P_{ij}) using randomness ρ. By construction, the function g is indeed a sum of 2-local functions, as required. The output of f can be recovered from the output of g by applying the ARE decoder Dec of Π. Intuitively, a set of corrupted parties can learn nothing (given their inputs, randomness, and the output of g) beyond the output of f because the honest parties contribute secret randomness to the ARE message of each virtual party that involves at least one honest party. Since the inputs of these virtual parties determine all inputs, the residual function of f' with these inputs fixed is determined by the output $f(x_1, \ldots, x_n)$. Finally, we note that by a known completeness results [ABG+20], applying the above transformation for to a constant-size 3-party function suffices.

3 Additive Randomized Encoding: Definitions and Properties

Here we define Additive Randomized Encoding (ARE), considering both information-theoretic and computational security, both with and without robustness. Below we define the ARE syntax in the most general case, where the function to compute and the group over which the encodings are added are parameterized by several parameters. In the sequel, not all the parameters will be relevant in all the settings, and we often omit some of them (e.g., in Section 4, we consider information-theoretic AREs, in which case setup algorithm, public parameters etc. are not needed, and functions are often defined over a finite domain).

Definition 3.1 (ARE Syntax)**.** *Let $f : (\{0,1\}^*)^* \to \{0,1\}^*$ be a multiparty function. An ARE scheme for f is a triple of algorithms $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^n, 1^\ell) \to \mathsf{pp}$ *is a PPT* setup algorithm *that, given security parameter λ, number of parties n, and input length ℓ, generates public parameters* pp*. The public parameters include λ, n, ℓ, and an explicit description of an Abelian group $\mathbb{G}$. They can also include some randomness, such as random generators of $\mathbb{G}$ and/or a common reference string (CRS). We will sometimes eliminate* Setup *and consider $\mathbb{G}$ as being fixed.*
- $\mathsf{Enc}(\mathsf{pp}, i, x_i) \to \hat{x}_i$ *is a PPT* encoding algorithm *that maps an input x_i of party i to a group element from $\mathbb{G}$. We refer to $\hat{x}_i$ as the encoding of x_i, or the ARE message of party i.*
- $\mathsf{Dec}(\mathsf{pp}, \hat{y}) \to y$ *is a PPT* decoding algorithm *that maps a group element $\hat{y} \in \mathbb{G}$ to an output y.*

Π *is correct, with a possible error of $\epsilon = \epsilon(\lambda)$, if for all λ, n, ℓ, and $x_1, \ldots, x_n \in \{0,1\}^\ell$:*

$$\Pr\left[\begin{array}{ll} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n, 1^\ell); & \\ \hat{x}_i \leftarrow \mathsf{Enc}(\mathsf{pp}, i, x_i); & : \mathsf{Dec}(\mathsf{pp}, \hat{y}) = f(x_1, \ldots, x_n) \\ \hat{y} = \sum_{i=1}^n \hat{x}_i & \end{array}\right] \geq 1 - \epsilon(\lambda),$$

where summation is taken over the group $\mathbb{G}$ specified by pp*. In the statistical and computational settings, we require by default that ϵ is negligible in the security parameter.*

We will often consider functions f for which n and/or ℓ are fixed. In such cases, these parameters will be omitted. Also, in some cases we do not need the setup procedure at all, and in the perfect security setting we do not have a security parameter.

Remark 3.2 (Sending messages to the evaluator)**.** *Note that the encoded input could be a vector, where we use different slots for different purposes. The ARE group in this case is the direct product of the groups in all the slots.*

This syntax allows parties to directly send messages to the evaluator, by allocating a slot in the vector to one party, where that party puts the message and all other parties put zeros. We use the shorthand "party P_i sends $(x_i; y_i)$" to mean that x_i is added to the x'es of all the other parties, and y_i is sent directly to the evaluator.

3.1 ARE Security

Our basic security notion, without robustness, asserts that $\hat{y}$ can be simulated given access to $f(x_1, \ldots, x_n)$. For any λ, n, ℓ and $x_1, \ldots, x_n \in \{0,1\}^\ell$, let us denote by $\Pi\big(1^\lambda, x_1, \ldots, x_n\big)$ the output of the process:

$$\Pi\big(1^\lambda, x_1, \ldots, x_n\big) \;:=\; \left\{ \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n, 1^\ell); \hat{x}_i \leftarrow \mathsf{Enc}(\mathsf{pp}, i, x_i); \hat{y} = \sum_{i=1}^n \hat{x}_i; \\ \text{output } (\mathsf{pp}, \hat{y}) \end{array} \right\}.$$

We often omit some of these parameters, if they are irrelevant in a given context.

Definition 3.3 (ARE Security). *An ARE scheme* $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ *for* $f : (\{0,1\}^*)^* \to \{0,1\}^*$ *as in Definition 3.1, is said to be secure if there exists a randomized algorithm* Sim*, called a simulator, such that:*

Perfect security. *For all* n, ℓ *and* $x_1, \ldots, x_n \in \{0,1\}^\ell$, $\mathsf{Sim}\big(1^n, 1^\ell, f(x_1, \ldots, x_n)\big) \equiv \Pi\big(x_1, \ldots, x_n\big)$.

Statistical security. *For some negligible function* $\delta(\cdot)$*, it holds for all* λ, n, ℓ *and* $x_1, \ldots, x_n \in \{0,1\}^\ell$*, that*

$$SD\left(\mathsf{Sim}\big(1^\lambda, 1^n, 1^\ell, f(x_1, \ldots, x_n)\big),\ \Pi\big(1^\lambda, x_1, \ldots, x_n\big)\right) \le \delta(\lambda),$$

where $SD(\cdot, \cdot)$ *is the statistical distance.*

Computational security. Sim *is a PPT algorithm, and for all* λ, n, ℓ *and* $x_1, \ldots, x_n \in \{0,1\}^\ell$, $\mathsf{Sim}\big(1^\lambda, 1^n, 1^\ell, f(x_1, \ldots, x_n)\big)$ *and* $\Pi\big(1^\lambda, x_1, \ldots, x_n\big)$ *are computationally indistinguishable.*

Robustness. The above definition only considers security against the *external* evaluator who only sees the sum of the ARE encodings. When the evaluator may collude with a subset of the parties, we need a stronger notion of *robust ARE*.

It is easy to see that a collusion between the evaluator and some parties can get the sum of encodings of the other (honest) parties by subtracting out from $\hat{y}$ the encodings of the colluding parties. Below we denote, for any subset of honest parties $H \subset [n]$,

$$\Pi_H\big(1^\lambda, x_1, \ldots, x_n\big) \;:=\; \left\{ \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n, 1^\ell); \hat{x}_i \leftarrow \mathsf{Enc}(\mathsf{pp}, i, x_i); \hat{y}_H = \sum_{i \in H} \hat{x}_i; \\ \text{output } (\mathsf{pp}, \hat{y}) \end{array} \right\}.$$

Clearly, a collusion of the evaluator with the parties in $[n] \setminus H$ necessarily gets access to the *residual function* of the honest parties in H. In the definition below for robust ARE, the simulator will therefore get access not just to $f(x_1, \ldots, x_n)$, but to the entire residual function.

For any n-party function f, subset $H \subset [n]$, and inputs $\mathbf{x} = (x_i : i \in H) \in (\{0,1\}^\ell)^{|H|}$, the residual function defined by $H, \mathbf{x}$ is the following function on $m = n - |H|$ inputs:

$$f_{H,\mathbf{x}}(w_1, \ldots, w_m) = f(z_1, \ldots, z_n), \text{ where } z_i = \begin{cases} x_i & \text{if } i \in H \\ w_{j_i} & \text{if } i \notin H \end{cases},$$

where j_i, for $i \notin H$, is the index of i in the set $[n] \setminus H$.

Definition 3.4 (Robust ARE)**.** *An ARE scheme* $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ *for* $f : (\{0,1\}^*)^* \to \{0,1\}^*$ *as in Definition 3.1, is said to be* robust *if there exists a simulator* Sim *with access to a residual-function oracle, such that:*

Perfect robustness. *For all* n, ℓ*,* $H \subset [n]$*, and inputs* $\mathbf{x} = (x_i : i \in H)$*,* $\mathsf{Sim}^{f_{H,\mathbf{x}}}\big(1^n, H, 1^\ell\big) \equiv \Pi_H\big(\mathbf{x}\big)$*.*

Statistical robustness. *For some negligible function* $\delta(\cdot)$*, it holds for all* λ, n, ℓ*,* $H \subset [n]$*, and inputs* $\mathbf{x} = (x_i : i \in H)$*, that*

$$SD\left(\mathsf{Sim}^{f_{H,\mathbf{x}}}\big(1^\lambda, 1^n, H, 1^\ell\big),\ \Pi_H\big(1^\lambda, \mathbf{x}\big)\right) \le \delta(\lambda),$$

where $SD(\cdot,\cdot)$ *is the statistical distance.*

Computational simulation-robustness. Sim *is a PPT algorithm, and for all* λ, n, ℓ*,* $H \subset [n]$*, and inputs* $\mathbf{x} = (x_i : i \in H)$*,* $\mathsf{Sim}^{f_{H,\mathbf{x}}}\big(1^\lambda, 1^n, H, 1^\ell\big)$ *and* $\Pi_H\big(1^\lambda, \mathbf{x}\big)$ *are computationally indistinguishable. We will also consider the weaker notion of computational VBB-robustness, where the order of quantifiers is reversed: for every efficient Boolean distinguisher* $\mathcal{A}$ *there is an efficient simulator* Sim *such that* $\mathcal{A}$ *has a negligible advantage distinguishing between* $\mathsf{Sim}^{f_{H,\mathbf{x}}}$ *and* Π_H*.*

Indistinguishability robustness. Sim *is an unbounded algorithm, and for all* λ, n, ℓ*,* $H \subset [n]$*, and inputs* $\mathbf{x} = (x_i : i \in H)$*,* $\mathsf{Sim}^{f_{H,\mathbf{x}}}\big(1^\lambda, 1^n, H, 1^\ell\big)$ *and* $\Pi_H\big(1^\lambda, \mathbf{x}\big)$ *are computationally indistinguishable.*

Note that the indistinguishability variant of robustness can be equivalently defined by requiring that for any H, if $\mathbf{x}$ and $\mathbf{x}'$ induce the same residual function (namely, $f_{H,\mathbf{x}} \equiv f_{H,\mathbf{x}'}$) then their partial sums $\sum_{i\in H} \hat{x}_i$ and $\sum_{i\in H} \hat{x}'_i$ are computationally indistinguishable, even given pp.

Remark 3.5 (On the different notions of robust computational ARE)**.** *In the 2-party case (*$n = 2$*), the different notions of robust ARE are analogous to corresponding notions of obfuscation. In fact, a robust ARE for a* universal *function* f *implies obfuscation of the corresponding function class. Similarly to obfuscation, the simulation and VBB variants are generally impossible to realize [BGI+12]. The VBB variant can potentially be realized for simple but nontrivial classes of functions, such as evasive functions [BBC+14]. In contrast, the simulation variant is only meaningful in idealized models, such as the ideal obfuscation model [JLLW22].*

We will be particularly interested in "constant-size" functions f*, for which both* n *and* ℓ *are bounded. While there is no meaningful notion of obfuscation for constant-size functions, here we get a meaningful notion, which is stronger than the non-robust notion. (This is similar to the robust non-interactive MPC model from [BGI+14].) For such constant-size* f*, all the above notions of robustness are equivalent, and we refer to the indistinguishability variant by default.*

Remark 3.6 (On separating robust ARE from standard ARE)**.** *Let* $f(x,y)$ *be a constant function, for* $x, y \in \{0,1\}$*. Consider the following ARE for* f *over* $\mathbb{Z}_m$*: input* x *is encoded by* x *itself, while* y *is encoded by* $r \in_R \mathbb{Z}_m$*. This ARE satisfies the standard notion of security, as the sum of encodings is random in* $\mathbb{Z}_m$*, but the evaluator together with the second party learn* x *(which they should not).*

3.2 Basic Properties of ARE

We note that if we have ARE schemes for two functions f, f' defined over the same set of inputs, then we also have an ARE for the function $g(\mathbf{x}) = (f(\mathbf{x}), f'(\mathbf{x}))$. This is analogous to the standard concatenation property for randomized encoding of functions.

Claim 3.7 (cf. [AIK14], Fact 3.3)**.** *Let f, f' be two n-party function. If both functions have perfect (alternatively, statistical or computational) secure/robust ARE schemes, then the concatenation function $g = (f, f')$ also has an ARE of the same type.*

Proof. An ARE Π'' for g, is obtained by concatenating the two ARE schemes for f, f': The public parameters are concatenated, and so are the encodings. The evaluator decodes each part separately. The various types of security are easy to verify. □

3.2.1 ARE "over the integers"

It is sometimes convenient to convert an ARE for some function f from one group to another. For that purpose, it is convenient to talk about ARE schemes over the ring of integers $\mathbb{Z}$. While the syntax in Definition 3.1 requires a finite group, it is often easy to project $\mathbb{Z}$ down to some $\mathbb{Z}_t$ for large enough t, such that the summation of encodings (almost) never triggers modular reduction. We refer to such ARE scheme with no modular reduction as being "over the integers."

Conveniently, we can convert ARE over $\mathbb{Z}_m$ to one over the integers, using the standard technique of adding a sufficiently large random multiple of m.

Claim 3.8 (cf. [AIK14], Claim 5.3)**.** *If f has a statistical/perfect secure ARE scheme over $\mathbb{Z}_m$, then it has a statistical ARE over the integers.*

Proof. The scheme over the integers has $\mathsf{Enc}'_i(x_i) = \mathsf{Enc}_i(x_i) + r_i \cdot m$ (over the integers), where $r_i \in_R [0, \mu - 1]$ for some parameter μ.[3] The decoder just reduce everything modulo m and runs the original decoder. For security, we note that if the original scheme has n parties and statistical security upto δ, the new one will have statistical security upto at most $\delta + n/\mu$. Choosing μ sufficiently large gives what we need. □

Note that the proof in fact uses only a finite portion of the integers, so we can again view the resulting scheme as being over some large enough $\mathbb{Z}_t$.

This can be useful for proving impossibility results: For example, suppose that we could rule out statistical ARE schemes for a certain function modulo any prime. Then we can use this to also rule out ARE over $\mathbb{Z}_m$ for non-prime moduli: if there were such a scheme over $\mathbb{Z}_m$, we could use this to convert it to an ARE over $\mathbb{Z}_t$ for a prime t sufficiently larger than m. (See for example the claim in Section 4.1.)

4 Information-Theoretic ARE

We start with few examples of information-theoretic AREs for some simple (but useful) functions. We use D_i to denote the input domain of the i-th party, and $\mathsf{Enc}_i = \mathsf{Enc}(\perp, i, \cdot)$ to denote its encoder.

[3] In the context of information-theoretic AREs it is often convenient to replace the notation $\mathsf{Enc}(\mathsf{pp}, i, x_i)$ by $\mathsf{Enc}_i(x_i)$.

Example 4.1. (ARE for the OR function.) *Here $D_i = \{0, 1\}$, for $i \in [n]$, and all encoders Enc_i are identical: on input $x_i = 0$ output $g_i = 0$, while on input $x_i = 1$ output $g_i \in_R \mathbb{G}$. The decoder Dec on input $g = 0$ outputs 0 while on any other $g \in \mathbb{G}$ it outputs 1. Observe that when $OR(\mathbf{x}) = 0$, all g_i's are 0 and so is their sum g. While if $OR(\mathbf{x}) = 1$, at least one g_i is random in $\mathbb{G}$ and therefore so is g. Hence, correctness holds except with probability $1/|\mathbb{G}|$ (if $OR(\mathbf{x}) = 1$ but still $g = 0$). security, on the other hand, is perfect: there is a requirement only for inputs $\mathbf{x}$ where $OR(\mathbf{x}) = 1$, and for those inputs g is uniformly random in $\mathbb{G}$ which is easily simulatable.*

An ARE for the function AND is constructed in a similar way.

Example 4.2. (ARE for the MAX function.) *Here, for all $i \in [n]$, the input domain is $D_i = [M]$, for some integer M. For some group $\mathbb{G}_0$ our ARE will be over $\mathbb{G} = (\mathbb{G}_0)^{M-1}$. Specifically, all encoders Enc_i are identical: on input $x_i \in D_i$ output a length $M - 1$ vector with uniform and independent random elements from $\mathbb{G}_0$ in each of the first $x_i - 1$ entries and 0's in all other entries. The decoder Dec on input $g \in \mathbb{G}$ outputs the index of the first coordinate of g which is 0 (or M if no coordinate is 0). Observe that if $MAX(\mathbf{x}) = m$, then the first $m - 1$ entries of g are random (because so is any g_i that corresponds to $x_i = m$) and the other entries are all 0's. This implies perfect security. Correctness error may happen if one of the random entries in g turns out to be 0. This happens with probability $1/|\mathbb{G}_0|$ and so the ARE is correct with probability $\geq (1 - 1/|\mathbb{G}_0|)^{M-1}$. Choosing $\mathbb{G}_0$ of appropriate size will guarantee a small error. We note that this ARE can be viewed as $M - 1$ invocations of the ARE for OR (on carefully selected inputs) but for the purpose of example we use a direct approach.*

Example 4.3. (ARE for an "equality"-type function.) *Consider the following two-party function $Equal$. The input domains are $D_1 = D_2 = [M]$, for some integer M. Define $Equal(x_1, x_2) = 0$ if $x_1 \neq x_2$ and $Equal(x_1, x_2) = v$ if $x_1 = x_2 = v$. For some group $\mathbb{G}_0$ our ARE will be over $\mathbb{G} = (\mathbb{G}_0)^M$. The encoders Enc_i are identical: on input $x_i \in [M]$ output a length M vector with 0 at the i-th entry and random elements from $\mathbb{G}_0$ in all other entries. The decoder Dec on input $g \in \mathbb{G}$ outputs the first coordinate of g which is 0 or 0 if no such coordinate exists. Observe that if $x_1 \neq x_2$ then all entries of g are random, while if $x_1 = x_2 = v$ then we must have a 0 in entry v and all other entries are random. Again, this implies perfect security. Correctness error may happen when $x_1 \neq x_2$ but one (or more) of the entries are 0 (of $\mathbb{G}_0$) by chance, or if $x_1 = x_2 = v$ but, in addition to the v-th entry, other random entries in g turn out to be 0. Therefore, the ARE is correct with probability $\geq (1 - 1/|\mathbb{G}_0|)^M$.*

Note that the examples above all refer to symmetric functions and hence it is natural to also have a "symmetric" ARE, where the encoders corresponding to all inputs are identical.

Example 4.4. (ARE for multiplication modulo a prime.) *Let $D_i = \mathbb{Z}_p$, for some prime p and let $Mult_p(\mathbf{x}) = \Pi_{i=1}^n x_i \bmod p$. Consider the following ARE over $\mathbb{G} = \mathbb{Z}_m \times \mathbb{Z}_{p-1}$. Each Enc_i works as follows: on input $x_i = 0$ it outputs (a_i, b_i), where $a_i \in_R \mathbb{Z}_m, b_i \in_R \mathbb{Z}_{p-1}$, and for input $x_i \neq 0$ it outputs $(0, \log_g x_i)$, where g a generator of $\mathbb{Z}_p$. The decoder Dec on input $(a, b) \in \mathbb{G}$ outputs 0 if $a \neq 0$ and $g^b \bmod p$ if $a = 0$. Observe that if any of the x_i's is 0 (i.e., $Mult_p(\mathbf{x}) = 0$) then the output (a, b) is random in $\mathbb{G} = \mathbb{Z}_m \times \mathbb{Z}_{p-1}$, while if $Mult_p(\mathbf{x}) = s \neq 0$ then the output is $(0, \log_g s)$. This implies perfect security. Correctness error may happen when $Mult_p(\mathbf{x}) = 0$ but $a = 0$ by chance, which happens with probability $1/m$.*

ARE for Multiplication modulo general integer m (i.e., potentially non-prime) is a corollary of the ARE for capped sum, presented in the next section.

We observe that each of the above AREs is also robust. The following theorem summarizes the above examples:

Theorem 4.5. *There exist statistically-correct (over sufficiently large groups), perfectly-secure, robust AREs for the functions OR, MAX, Equal and $Mult_p$. Concretely, let $\epsilon > 0$ be the desired bound on the correctness error, then our ARE for OR requires a group $\mathbb{G}$ of size at least $1/\epsilon$, our ARE for MAX over the domain $[M]$ requires a group $\mathbb{G} = (\mathbb{G}_0)^{M-1}$ with $\mathbb{G}_0$ of size $\Omega(M/\log 1/(1-\epsilon))$, our ARE for Equal over the domain $[M]$ requires a group $\mathbb{G} = (\mathbb{G}_0)^M$ with $\mathbb{G}_0$ of size $\Omega(M/\log 1/(1-\epsilon))$, and our ARE for $Mult_p$ requires a group $\mathbb{G} = \mathbb{Z}_m \times \mathbb{Z}_{p-1}$ with $m \geq 1/\epsilon$.*

4.1 ARE for Capped Sum

In this section, we consider the capped-sum function that returns the sum of inputs $x_1, \ldots, x_n$, unless the sum is larger than some cap parameter θ, in which case the function returns θ. For simplicity, we assume in this section that the input domains are $D_i = \{0, 1\}$, though our results can be extended to larger domains (see Remark below). The special cases of $\theta = 1$ and $\theta = n$ are already covered by OR_n and SUM_n, respectively.

The capped-sum function has several motivations. On the technical side, it serves as a building block in the construction of other protocols (see below). However, it is also motivated as a stand-alone functionality. For example, consider a "whistle blowing" scenario in which n parties are given the opportunity to complain about something/somebody but, for privacy considerations, the number of complaints will be exposed only if it is above some threshold t. This can be achieved via a capped-sum protocols, where each party inputs 0 for "complain" and 1 for non-complain. The cap will be $\theta = n - t$. If there are at most t complaints then the output will be θ. However, if more than t complaints are cast then the output is some $y < \theta$ (which implies $n - y$ complaints).

Claim 4.6. *Let $f : \{0,1\}^n \to \{0, 1, \ldots, \theta\}$ be the capped sum function with cap θ. Then, there exists a statistically-secure, robust ARE for f over $\mathbb{G} = (\mathbb{F}_p)^{\theta^2}$.*

Proof. The encoding algorithm Enc_i on input $x_i = 0$ outputs an all-0 matrix M_i of dimension $\theta \times \theta$, and on input $x_i = 1$ it outputs M_i which is a random $\theta \times \theta$ rank-1 matrix, with entries in $\mathbb{F}_p$ (selected by choosing random, length θ vectors u_i, v_i over $\mathbb{F}_p$ and setting $M_i = u_i \times v_i^T$). Denote $M = \sum_{i=1}^n M_i$. The decoder on input M outputs $\mathrm{rank}_{\mathbb{F}_p}(M)$. Obviously, this rank is in the range $\{0, 1, \ldots, \theta\}$.

If $\sum_{i=1}^n x_i = s$ then M is the sum of s rank-1 random matrices (in the special case $s = 0$, the input $\mathbf{x}$ is uniquely determined; hence, we can assume $s \geq 1$). For correctness, we argue that with high probability $\mathrm{rank}_{\mathbb{F}_p}(M) = \min(s, \theta)$. For this, let A be a $\theta \times s$ matrix whose columns are the corresponding u_i's, and let B be a $s \times \theta$ matrix whose rows are the corresponding v_i's. It follows that $M = A \cdot B$. Next, we argue that with high probability (depending on p) each of A, B has full rank $\min(\theta, s)$ (Lemma 4.7 below) and that the rank of $A \cdot B$ is also $\min(\theta, s)$ (Lemma 4.8 below). The correctness, with high probability, follows.

For security, if $f(\mathbf{x}) = s < \theta$ then M is always the sum of s rank-1 matrices (for such inputs security is perfect). The case $f(\mathbf{x}) = \theta$ is more involved, as the larger $\sum x_i$ is the larger the probability that $M = A \cdot B$ has full rank, in which case M is a random such matrix. It follows from Lemma 4.8 that the difference is small. □

Lemma 4.7. *Let A be a randomly selected $m \times n$ matrix over $\mathbb{F}_p$ and assume, without loss of generality, that $m \leq n$. Then, A has full rank (i.e., $\min(m, n)$) with probability $> (p-2)/(p-1)$.*

Proof. Select A row-by-row. A has full rank iff each row a_i is independent of the previous rows. The first $i-1$ independent rows span p^{i-1} vectors and so the probability that a_i is not one of them is $(p^n - p^{i-1})/p^n = 1 - 1/p^{n+1-i}$. Hence, the probability that all m rows are independent is $\Pi_{i=1}^{m}(1 - 1/p^{n+1-i}) \geq \Pi_{i=1}^{n}(1 - 1/p^i)$. To bound this expression, it is convenient to consider the complement event, where A is singular. This happens with probability at most $\sum_{i=1}^{m} 1/p^{n+1-i} < \sum_{i=1}^{\infty} 1/p^i = 1/(p-1)$. The claim follows. □

Lemma 4.8. *Let B be a full rank matrix of dimension $s \times \theta$ over $\mathbb{F}_p$. Let A be a random matrix of dimension $\theta \times s$ over $\mathbb{F}_p$. Then, rank$(AB) = \min(s, \theta)$ with high probability.*

Proof. Case 1: $s \leq \theta$. In this case, B has a subset of s columns that form an $s \times s$ invertible matrix B'. By Lemma 4.7, A has rank s with probability $\geq (p-2)/(p-1)$, hence $A \cdot B'$ also has rank s and, moreover, it is a submatrix of $A \cdot B$ which therefore also has rank s.
Case 2: $s > \theta$. By Lemma 4.7, A has rank θ with probability $\geq (p-2)/(p-1)$, in which case it has a subset of θ rows that form a $\theta \times \theta$ invertible matrix A'. Since B has rank θ then so does $A' \cdot B$ and also $A \cdot B$. □

We note that, unlike the previous examples, here we only get an ARE with *statistical security.* This relaxation is shown below to be necessary.

Remark 4.9. *To deal with capped-sum over a larger domain $\{0, 1, \ldots, m\}$, we can choose M_i as a random matrix of rank x_i or, to make the analysis similar to the binary case, as the sum of x_i rank-1 random matrices.*

Capped sum with payloads. Next, we describe an extension of the capped sum functionality, that we sometimes refer to as "capped sum with payloads". Concretely, each party has a pair of inputs (x_i, y_i). If the capped-sum of $\mathbf{x}$ is below the threshold θ then the ARE reveals this capped-sum along with $\sum_{i=1}^{n} y_i$, while if the capped-sum is at least θ then nothing about $\mathbf{y}$ is revealed. More generally, we can replace $\sum_{i=1}^{n} y_i$ by any function $f(y)$ for which an ARE over some $\mathbb{Z}_m$ exists; we denote this functionality CS_f.

Claim 4.10. *Let f be a function with an ARE over $\mathbb{Z}_m$. Then, there is a statistical ARE for CS_f over $\mathbb{Z}_{m'}$, for sufficiently large m'.*

Proof. The idea is to reduce this problem to the standard capped-sum, by increasing the range from $\{0, 1, \ldots \theta\}$ to $\{0, 1, \ldots \theta \cdot \beta\}$, where $\mathbb{Z}_\beta$ is the group to which we can transform the ARE for f, as described in Section 3.2.1, and is sufficiently large to make sure that there no modular reduction occurs. Specifically, each x_i is first multiplied by β and then we add to it $\mathsf{Enc}_i(y_i) \in \mathbb{Z}_\beta$ (where Enc_i is the encoder described in Section 3.2.1) to get a new input z_i for capped-sum (with cap $\theta \cdot \beta$).

If the sum of x_i's is at least θ then the sum of z_i's is at least $\theta \cdot \beta$ and so capped-sum reveals no additional information on the inputs. If the sum of x_i's is some $s < \theta$ then the sum of z_i's is $< \theta \cdot \beta$ and more precisely, it is $s \cdot \beta$ (from the x_i's) plus the sum of ARE encodings in $\mathbb{Z}_\beta$ (where here we use the property of this ARE that no modular reduction occurs). This allows for decoding s and, from the sum of encodings, decoding also the value $f(\mathbf{y})$ (but no other information about $\mathbf{y}$). □

Example 4.11. (ARE for Multiplication over $\mathbb{Z}_m$.) *Let $D_i = \mathbb{Z}_m$, for some integer m and let $Mult_m(\mathbf{x}) = \Pi_{i=1}^{n} x_i \bmod m$. Our ARE for $mult_p$ reduced multiplication to summation of $\log x_i$*

(in the case that the product is not 0). While (discrete) log is defined in $\mathbb{Z}_p$ it is not defined in $\mathbb{Z}_m$, hence extra care is required.

We start with the case that $m = p^e$. We write each x_i as $x_i = p^{e_i} \cdot y_i$, where y_i co-prime with p, and encode it into the pair (e_i, b_i), where $b_i = \log_g y_i$ for g a generator of the multiplicative group $\mathbb{Z}^*_{p^e}$ (for convenience we view a 0 input as p^e). Let $e' = \sum_{i=1}^n e_i$. If $e' \geq e$ then $Mult_m(\mathbf{x}) = 0$; otherwise, it is $\Pi_{i=1}^n x_i = p^{e'} \cdot \Pi_{i=1}^n y_i = p^{e'} \cdot g^{\sum_{i=1}^n b_i}$. Note, that if $e' \geq e$ then its exact value should not be revealed. This is achieved by computing e' as the capped sum of $e_1, \ldots, e_n$. Moreover, if this capped sum equals e, the output is 0 and the ARE should not reveal $\sum_{i=1}^n b_i$. This is immediately solved via the capped sum with payloads ARE, described above (Claim 4.10).

Finally, we consider the case of general m; that is, $m = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$. In this case, the ARE is constructed by just applying, for each $i \in [k]$, the ARE for $Mult_{p_i^{e_i}}(\mathbf{x})$, which uniquely determine $Mult_m(\mathbf{x})$ via the Chinese Remainder Theorem (CRT). Whenever all the e_i's equal 1, we can use the much more efficient $Mult_{p_i}$ encodings.

4.2 Negative Results for Perfectly Secure ARE

In this section we present some impossibility results for Information-theoretic AREs. These results complement the positive results above, and should also be contrasted with the strong positive results in the *computational* case, presented in the next section.

Towards these negative results, we start by presenting some tools. We first review some facts about the Discrete Fourier Transform of functions representing probability distributions. Then we introduce a new notion that we call "Vector Multiplication Programs" (VMP) that is central to our negative results, and is motivated by the connection to Fourier Transform, and connect it to ARE. Finally we use this connection in to prove negative results for *perfect* information-theoretic ARE of various functions. Proving negative results for *statistical* AREs, remains as an intriguing open problem.

4.2.1 Discrete Fourier Transform for Distributions

The notion of ARE relies on distributions over $\mathbb{G}$, representing the randomized mappings from each input x_i to its encoding; intuitively, we ask that the two distributions $\sum \mathsf{Enc}(x_i)$ and $\sum \mathsf{Enc}(y_i)$ are "close" if $f(\mathbf{x}) = f(\mathbf{y})$ and are "far" if $f(\mathbf{x}) \neq f(\mathbf{y})$. To analyze such probability distributions, it is useful to view them as functions $p_{x_i} : \mathbb{G} \to \mathbb{R}$ (assigning to each element of the group $\mathbb{G}$ its probability to be chosen as the encoding of a certain value x_i), where the group $\mathbb{G}$ is assumed to be finite and Abelian.

The *Discrete Fourier Transform* (DFT) is a "change of basis" that gives a way to express each such function in an orthonormal basis that have convenient properties.

The DFT of any function $f : \mathbb{G} \to \mathbb{C}$ is defined, using a set of basis functions called *characters*. For groups of the form $\mathbb{Z}_m$, the standard characters are $\chi_j : \mathbb{Z}_m \to \mathbb{C}$, defined by $\chi_j(x) = \omega^{jx}$, where ω is the m-th root of unity and $j \in \{0, \ldots, m-1\}$. The change of basis from f to its Fourier representation $\widehat{f}$ can be obtained by viewing f as a length m vector and writing $\widehat{f} = V \cdot f$, where V is an $m \times m$ (normalized) Vandermonde matrix with $V_{i,j} = \omega^{ij}/\sqrt{m}$, for $0 \leq i, j < m$.[4] If $\mathbb{G} = \mathbb{Z}_m^n$ the characters are obtained by products of the above; that is, for $\alpha \in \{0, \ldots, m-1\}^n$, we have

[4] The notation $\widehat{f}$ is a standard notation for the Fourier representation of f and is used only in Section 4.2 of this paper. It is unrelated to the notation of encoding (e.g., $\hat{x}_i$ denotes the encoding of x_i) that we use in other parts of the paper, and is standard in the randomized-encoding literature.

$\chi_\alpha : \mathbb{Z}_m^n \to \mathbb{C}$ defined by $\chi_\alpha(x) = \Pi_{j=1}^n \chi_{\alpha_j}(x_j)$. The general case, where the finite Abelian group $\mathbb{G}$ is expressed as a product of cyclic prime order groups, the characters are similarly obtained via product of characters of the corresponding cyclic groups.

If $p, q : \mathbb{G} \to \mathbb{R}$ represent two probability distributions, corresponding to two random variable X, Y over $\mathbb{G}$, then their *convolution*, denoted by $p * q$, represents the distribution of the random variable $X + Y$.[5] That is, for all $z \in \mathbb{G}$, we have $(p * q)(z) = \sum_{x \in \mathbb{G}} p(x) \cdot q(z - x)$.

The next theorem (see, e.g., [O'D14, Thm 8.60]) is extremely useful, stating that in the Fourier representation, the coefficients of the convolution $p * q$ can be obtained simply by multiplying the corresponding coefficients of $\widehat{p}, \widehat{q}$

Proposition 4.12 (Convolution Theorem.)**.** *Let $p, q : \mathbb{G} \to \mathbb{R}$ be two distributions. Then, for all α, $\widehat{p * q}(\alpha) = \widehat{p}(\alpha) \cdot \widehat{q}(\alpha)$.*

4.2.2 Vector Multiplication Programs

In this section, we present a model that we term *Vector Multiplication Programs* (VMP), that we will use for proving our negative results for ARE. For intuition, consider a two-argument function $f : D_1 \times D_2 \to R$ and assume that we have an ARE over a group $\mathbb{G}$ for f. Then, we can associate with each value $x \in D_1$ a probability distribution $p_x : \mathbb{G} \to \mathbb{R}$, induced by the encoder Enc_1 and, similarly, with each value $y \in D_2$ a probability distribution $q_y : \mathbb{G} \to \mathbb{R}$, induced by the encoder Enc_2. As explained above, the sum of these two random encodings, is distributed as the *convolution* $p_x * q_y$ and these distribution, over the different (x, y)'s should satisfy Correctness and Security, as required by the definition of ARE. Finally, we rely on the Convolution Theorem, that states that if we represent the probability distributions p_x, q_y using their Fourier representation $\widehat{p}_x, \widehat{q}_y$, then the Fourier representation of $p_x * q_y$ is simply the coordinate wise multiplication of $\widehat{p}_x, \widehat{q}_y$.

For convenience, we present a definition that corresponds to the simpler case of ARE with *perfect* security (but not necessarily perfect correctness). In this case, for any possible output $z \in R$, all inputs x, y such that $f(x, y) = z$ should have the *same* $p_x * q_y$ and hence $\widehat{p_x * q_y}$ is the same, and by the convolution theorem this is just $\widehat{p}_x \cdot \widehat{q}_y$. On the other hand, if $f(x, y) \neq f(x', y')$ then $p_x * q_y, p_{x'} * q_{y'}$ are "far" and $\widehat{p_x * q_y}, \widehat{p_{x'} * q_{y'}}$ are, at least, different. Formally,

Definition 4.13. *A* Vector Multiplication Program (VMP) *for a function $f : D_1 \times \ldots \times D_n \to R$ is a collection of vectors $\{v_{i,x_i}\}_{i \in [n], x_i \in D_i}$ from $\mathbb{C}^s$, for some length s, satisfying the following properties:*

- *Perfect security: If $\mathbf{x}, \mathbf{y}$ are such that $f(\mathbf{x}) = f(\mathbf{y})$ then $\odot_{i=1}^n v_{i,x_i} = \odot_{i=1}^n v_{i,y_i}$, where $\odot$ stands for coordinate-wise multiplication of the vectors.*
- *Weak correctness:*[6] *If $\mathbf{x}, \mathbf{y}$ satisfy $f(\mathbf{x}) \neq f(\mathbf{y})$ then $\odot_{i=1}^n v_{i,x_i} \neq \odot_{i=1}^n v_{i,y_i}$.*

In other words, there are distinct vectors $u_z \in \mathbb{C}^s$, corresponding to all possible output values $z \in R$, such that for all $\mathbf{x}$ such that $f(\mathbf{x}) = z$, we have $\odot_{i=1}^n v_{i,x_i} = u_z$.

[5] Actually, convolution can be defined not just for functions that correspond to distributions and also the theorem applies to the more general case, but in this paper we will only be interested in the restricted case of distributions.

[6] Since this definition is used for proving negative results, weakening the definition only makes the results stronger.

4.2.3 The Relation between VMPs and Perfectly Secure AREs

Next, we formalize the intuition presented in the previous section to show that if a function f admits a perfectly-secure ARE over $\mathbb{G}$ then it also has a VMP with corresponding parameters. Formally,

Theorem 4.14. *Let $\epsilon < 1/2$ and $f : D_1 \times \ldots \times D_n \to R$ be a function with an ϵ-correct, perfectly secure ARE over a group $\mathbb{G}$. Then, f admits a VMP, with vectors in $\mathbb{C}^{|\mathbb{G}|}$.*

Proof. Denote $s = |\mathbb{G}|$ and let $\mathsf{Enc}_1, \ldots, \mathsf{Enc}_n$ be the encoding algorithms of the ARE for f. Let $p_{i,x_i} : \mathbb{G} \to \mathbb{R}$ be the output distribution of $\mathsf{Enc}_i(x_i)$. Let $v_{i,x_i} = \widehat{p_{i,x_i}}$ in $\mathbb{C}^s$ be the Fourier representation of p_{i,x_i}. We next argue that these vectors form a VMP for f.

By perfect security of the ARE, we know that for all $\mathbf{x}$ where $f(\mathbf{x})$ is the same, say z, then $\sum_{i=1}^n \mathsf{Enc}_i(x_i)$ is identically distributed; denote this distribution by V_z and note that we can write V_z as the convolution $p_{1,x_1} * \ldots * p_{n,x_n}$. This implies, by the Convolution Theorem, that for all $\mathbf{x}$'s that are mapped to z we have the same $\widehat{p_{1,x_1}} \odot \ldots \odot \widehat{p_{n,x_n}}$, which implies, using the definition of the vectors v_{i,x_i} that for all the $\mathbf{x}$'s that are mapped to the same z we have that $\odot_{i=1}^n v_{i,x_i}$ is identical. A similar argument, using the ϵ-correctness of the ARE, shows that if $f(\mathbf{x}) \neq f(\mathbf{y})$ then $\odot_{i=1}^n v_{i,x_i}$ and $\odot_{i=1}^n v_{i,y_i}$ are different. □

Remark 4.15. *The ARE to VMP transformation above, is what we need for our negative results. Namely, we will show that certain VMPs do not exist and conclude that corresponding AREs cannot exist. It is possible to show certain transformations also from VMP to ARE. We note that, as far as we know, no such VMP to ARE transformation respects efficiency. To see this, consider the example of the two-party MAX function over $[M]$. A (perfectly correct) VMP for this function, can use vectors in $\{0,1\}^{M-1}$ as follows: for any $z \in [M]$ let $v_{1,z} = v_{2,z} = V_z = 0^{z-1}1^{M-z}$. For all x, y it follows that $v_{1,x} \odot v_{2,y} = V_{MAX(x,y)}$. On the other hand, the best ARE that we know for MAX (cf. Theorem 4.5) has exponential in M many encodings.*

4.2.4 Functions Admitting VMP

We start by considering two-argument boolean functions f (i.e., functions with range $R = \{0,1\}$). If we also have that the domains are of size 2 (i.e., $|D_1| = |D_2| = 2$), then all boolean functions are either isomorphic to XOR, or isomorphic to OR or constant and hence all have AREs (XOR and the constant functions in a trivial way, and OR as in Theorem 4.5). Next, we consider boolean functions with larger domains and show that except for in degenerate cases, no such function admits a VMP.

Lemma 4.16. *Let $f : D_1 \times D_2 \to \{0,1\}$. Assume that $|D_1| \geq 2, |D_2| \geq 3$ (alternatively, that $|D_1| \geq 3, |D_2| \geq 2$) and that f is non-redundant (i.e., there are no x, x' such that $f(x,y) = f(x',y)$ for all y, and no y, y' such that $f(x,y) = f(x,y')$ for all x). Then, there is no VMP for f.*

Proof. Assume for contradiction, that $\{v_x\}_{x \in D_1}$ and $\{w_y\}_{y \in D_2}$ are vectors in $\mathbb{C}^s$, for some s, that form a VMP for f. Further, denote by u_0, u_1, the distinct vectors in $\mathbb{C}^s$ such that, for all (x,y), if $f(x,y) = b$, for $b \in \{0,1\}$ then $v_x \odot w_y = u_b$. Since $u_0 \neq u_1$ then, for some coordinate j, we have $u_0^j \neq u_1^j$.

Let $x \in D_1$ be such that $f(x,\cdot)$ is not constant (such x exists, as otherwise all rows are identical). Since, $|D_2| \geq 3$, there are $b \in \{0,1\}$ and $y_1, y_2, y_3 \in D_2$ such that $f(x,y_1) = f(x,y_2) = b$

and $f(x, y_3) = 1 - b$. Hence, $v_x \odot w_{y_1} = v_x \odot w_{y_2} = u_b$ and, in particular,

$$v_x^j \cdot w_{y_1}^j = v_x^j \cdot w_{y_2}^j.$$

Since f is non-redundant, there is some $x' \in D_1$ such that $f(x', y_1) \neq f(x', y_2)$ which implies that $w_{y_1} \neq w_{y_2}$ and, moreover, since $u_0^j \neq u_1^j$, we have $w_{y_1}^j \neq w_{y_2}^j$. Combining this with the above equation, we get that $v_x^j = 0$. It follows that both u_0^j, u_1^j equal 0, contradicting the choice of j. $\square$

Combined with our ARE to VMP transformation, we get that no such function admits an ARE. This leads to characterization of two-party functions with IT-AREs as those that are "isomorphic" to a 2×2 boolean function:

Corollary 4.17. *Let $f : D_1 \times D_2 \to \{0,1\}$ be a Boolean function. Then, f has information theoretic perfectly-secure ARE iff $f(x, y) = g(f_1(x), f_2(y))$, where $f_1 : D_1 \to \{0,1\}, f_2 : D_2 \to \{0,1\}$ and $g : \{0,1\} \times \{0,1\} \to \{0,1\}$.*

Proof. Let f, g, f_1, f_2 be as in the claim. Use an ARE $(\mathsf{Enc}_1, \mathsf{Enc}_2, \mathsf{Dec})$ for g, that exist for all 2×2 Boolean functions. An ARE for f applies the encoder Enc_1 on $f_1(x)$, the encoder Enc_2 on $f_2(x)$, and uses Dec for decoding.

In the reverse direction, assume that there is an ARE for f. Then, there is an ARE for the non-redundant version of f, by using $f_1(x)$ to map every x to a canonical element of its equivalence class (say, the "first"), and similarly $f_2(y)$ to map every y to a canonical element of its equivalence class. By the above lemma, non-redundant Boolean functions have ARE only if $|D_1|, |D_2| \leq 2$. Hence f_1, f_2 are Boolean and f is of the desired form. $\square$

Next, we consider (multiparty) functions f with boolean inputs and outputs, and show a condition that rules outs the existence of a VMP for most such f's and, as a corollary, also perfectly-secure AREs for them. Clearly this condition does not hold for constant functions or functions isomorphic to either $XOR_{\leq n}$ or $OR_{\leq n}$.

Lemma 4.18. *Let $f : \{0,1\}^n \to \{0,1\}$. Assume that for some $i \in [n]$, some input value $b \in \{0,1\}$ for x_i and some output value $\alpha \in \{0,1\}$, there are $\mathbf{y}, \mathbf{y}', \mathbf{y}'' \in \{0,1\}^{n-1}$ (inputs for all other $n-1$ variables) such that*

$$f(b, \mathbf{y}) = \alpha, f(b, \mathbf{y}') = \alpha, f(b, \mathbf{y}'') = 1 - \alpha,$$

and

$$f(1-b, \mathbf{y}) = \alpha, f(1-b, \mathbf{y}') = 1 - \alpha, f(1-b, \mathbf{y}'') = 1 - \alpha,$$

Then, there is no VMP for f.

Proof. The proof follows similar ideas to the proof of the previous lemma. Let $u_0 \neq u_1 \in \mathbb{C}^s$ be the two target vectors that correspond to outputs 0 and 1, and let j be a coordinate where $u_0^j \neq u_1^j$.

Assume there is a VMP for f. Let $i, b, \alpha, \mathbf{y}, \mathbf{y}', \mathbf{y}''$, as guaranteed by the claim's assumption. It follows that $v_{i,b} \odot v_{\mathbf{y}} = v_{i,b} \odot v_{\mathbf{y}'}$ (where $v_{\mathbf{y}}$ stands for the product of the $n-1$ vectors corresponding to the bits of $\mathbf{y}$) and, in particular, $v_{i,b}^j \cdot v_{\mathbf{y}}^j = v_{i,b}^j \cdot v_{\mathbf{y}'}^j$. Moreover, $f(1-b, \mathbf{y}) \neq f(1-b, \mathbf{y}')$ and since $u_0^j \neq u_1^j$ we have $v_{\mathbf{y}}^j \neq v_{\mathbf{y}'}^j$ (as both are multiplied by the same $v_{i,1-b_i}^j$). It follows that $v_{i,b}^j = 0$. However, by a similar argument applied to $1-b, \mathbf{y}', \mathbf{y}''$ we will also get that $v_{i,1-b}^j = 0$ which (together with $v_{i,b}^j = 0$) contradicts the assumption that $f(b, \mathbf{y}') \neq f(1-b, \mathbf{y}')$. $\square$

Next, we turn to the case of the capped-sum function, which does admit a statistically secure ARE and rule out perfectly secure ARE for it. This follows directly from the following claim. In fact, we rule it out even in the case $n = 2$ and where the inputs come from a small domain.

Claim 4.19. *There is no VMP for the two-argument capped-sum function $f(x, y)$ over the domain $\{0, 1, 2\}$ and with cap=2.* [7]

Proof. Assume that such VMP exists, and denote the VMP vectors corresponding to x by v_0, v_1, v_2, the VMP vectors corresponding to y by w_0, w_1, w_2 and by u_0, u_1, u_2 the distinct target vectors corresponding to the outputs $0, 1, 2$ (respectively). Let j be such that $u_1^j \neq u_2^j$.

Following ideas from the above proofs, since $f(1, 1) = f(1, 2)$ but $f(0, 1) \neq f(0, 2)$ it follows that $v_1^j \cdot w_1^j = v_1^j \cdot w_2^j$ and $w_1^j \neq w_2^j$ and so $v_1^j = 0$. However, since $f(1, 0) \neq f(1, 1)$, we have $v_1 \odot w_0 \neq v_1 \odot w_1$, and by the choice of j this means $v_1^j \cdot w_0^j \neq v_1^j \odot w_1^j$. This contradict the conclusion that $v_1^j = 0$. □

4.2.5 A Negative Result for Statistically Secure ARE?

A natural next step is to extend the negative results above also to the statistical case. Proving such a negative result would also have implications in other domains (see Section 7 below). For example, it is possible to construct "best-possible information-theoretic" secure-MPC protocol for a function f (BIT-MPC, for short [HIKR18]) from an ARE scheme for f, so proving lower bounds on BIT-MPC protocols requires in particular lower bounds on statistically-secure ARE schemes. Unfortunately, so far we were not able to prove such negative results. In Appendix A we sketch one approach that we tried, using a relaxation of the connection to VMP schemes presented above, to try to rule out "well clustered" distributions that are close when $f(\mathbf{x}) = f(\mathbf{x}')$ but are far away when $f(\mathbf{x}) \neq f(\mathbf{x}')$. We were able to rule out such VMPs when "close" and "far away" are measured by the l_2 norm, but not when they are measured by statistical distance (l_1 norm) which is what we need.

5 Computational ARE from Bilinear Maps

Next we show that under standard hardness assumptions, any multi-party function admits a computationally secure ARE scheme (without robustness). To that end we show:

1. A pairing-based ARE scheme for the two-party equality function (Section 5.1);
2. A reduction of any two-party boolean function over a polynomial-size domain to equality, with application to computing the two-party string-oblivious-transfer function (Section 5.2);
3. An ARE scheme for every efficiently-computable multi-party function, using the two-party string-oblivious-transfer function together with garbled circuits (Section 5.3).

Alternatively, in Section 5.1.3 we modify slightly the equality-function scheme, to get directly a two-party string-oblivious-transfer ARE scheme with improved efficiency. The generic reduction from equality is still interesting, however, as there may be other implementations of equality from weaker/different hardness assumptions.

[7] In fact, the proof rules out even the case with $D_1 = \{0, 1\}, D_2 = \{0, 1, 2\}$.

5.1 A Pairing-Based Two-Party Equality Scheme

5.1.1 Background: Pairing Groups and Squaring XDH

To comply with our "additive RE" frame of mind, we use additive notations for our pairing-friendly groups. Otherwise, the notations below are similar to [BF03, Sec 3.1].

Parameters. A "Pairing parameter-generator" is an efficient (possibly randomized) procedure $\mathcal{G}$, taking as input the security parameter λ. It outputs a description of additive groups G_1, G_2, G_T of the same order q, distinguished generators $g_1 \in G_1$ and $g_2 \in G_2$, efficient addition/subtraction procedures for these groups, and an efficiently computable and nontrivial bilinear map $e : G_1 \times G_2 \to G_T$. A distinguished generator in G_T can be computed as $g_T := e(g_1, g_2)$.

We require that $q > 2^\lambda$, and assume below for simplicity that the order q of these groups is known and a prime (but our protocols can easily be adjusted to the unknown-order, non-prime case). Sampling random elements in these groups can be done by drawing $\rho \leftarrow \mathbb{Z}_q$ and computing $x := \rho \cdot g_*$. We denote $(q, G_1, G_2, G_T, g_1, g_2, g_T, e) \leftarrow \mathcal{G}(1^\lambda)$.

Also, our protocols work in either the symmetric case ($G_1 = G_2$) or the asymmetric case ($G_1 \neq G_2$). In the symmetric case we will still need to have $g_1 \neq g_2$, and we assume that it is hard to compute $DLOG_{g_1}(g_2)$.

Squaring XDH. Recall that the XDH assumption (for the asymmetric case $G_1 \neq G_2$) simply asserts that the standard decision Diffie-Hellman hardness assumption holds in G_1. Similarly, squaring-XDH asserts that decision-squaring-DH holds in G_1.

Definition 5.1 (Squaring XDH)**.** *The decision Squaring-XDH holds for a parameter-generator $\mathcal{G}$, if the following distribution ensembles are computationally indistinguishable: $SQ_\lambda := (\mathsf{pp}_\lambda,\ \rho \cdot g_1,\ \rho^2 \cdot g_1)$, and $U_\lambda := (\mathsf{pp}_\lambda,\ \rho \cdot g_1,\ \rho' \cdot g_1)$, where $\mathsf{pp}_\lambda = (q, G_1, G_2, G_T, g_1, g_2, g_T, e) \leftarrow \mathcal{G}(1^\lambda)$ and $\rho, \rho' \leftarrow \mathbb{Z}_q$.*

We now describe our ARE scheme for two-party equality. Security of this scheme can be reduced to a hardness assumption in paring groups that we call ASDH, which is weaker than Squaring XDH. (ASDH is implied by Squaring SDH, but it could plausibly hold also in the symmetric setting $G_1 = G_2$, where Squaring XDH is easy.) The two-party equality function over domain D is

$$f_{eq} : D \times D \to \{0,1\}, \quad f_{eq}(x_1, x_2) = \begin{cases} 1 & \text{if } x_1 = x_2, \\ 0 & \text{otherwise.} \end{cases}$$

For our pairing-based protocol we let $D = \{0,1\}^\ell$, and use $\mathcal{G}(1^{\max(\lambda, \ell+1)})$ to ensure that the order q of the pairing groups is more than $2^{\ell+1}$. We then use an injective embedding function $\mathsf{emb} : \{0,1\}^\ell \to \left[1, \frac{q-1}{2}\right]$ to embed the parties' inputs as integers between 1 and $\frac{q-1}{2}$. For example, $\mathsf{emb}(s) = \mathsf{bin}(s) + 1$, with $\mathsf{bin}(s)$ is the integer whose ℓ-bit binary expansion is s. Since it is injective then $x_1 = x_2$ if and only if $\mathsf{emb}(x_1) = \mathsf{emb}(x_2)$, and since the range is $\left[1, \frac{q-1}{2}\right]$ then $\mathsf{emb}(x_1) + \mathsf{emb}(x_2) \neq 0 \pmod q$ for any $x_1, x_2 \in \{0,1\}^\ell$. The protocol is described in Figure 1.

Correctness. Denote $\delta = \chi_1 - \chi_2$, so we have $e(y_1, y_2) = (\sigma_1 + \sigma_2)(\chi_1\sigma_1 - \chi_2\sigma_2) \cdot g_T = \big(\chi_1\sigma_1^2 - \chi_2\sigma_2^2 + (\chi_1 - \chi_2)\sigma_1\sigma_2\big) \cdot g_T = y_3 + \delta\sigma_1\sigma_2 \cdot g_T$. If $\delta \neq 0$ then equality only holds when $\sigma_1 = 0$ or $\sigma_2 = 0$, which happens with probability at most $2/q$.

Parameters: Parameter-generating procedure $\mathcal{G}$, security parameter λ, input length ℓ.

Setup$(1^\lambda, 1^\ell)$:

1. Let $\lambda' = \max(\lambda, \ell+1)$, set $\mathsf{pp} \leftarrow \mathcal{G}(1^{\lambda'})$ (recall that $q > 2^{\ell+1}$);
2. Let $\mathsf{emb} : \{0,1\}^\ell \to [1, \frac{q-1}{2}]$ be injective (e.g., $\mathsf{emb}(s) = \mathsf{bin}(s) + 1$).

Encoding, $\mathsf{Enc}(\mathsf{pp}, i, x_i)$:

3. Embed $\chi_i := \mathsf{emb}(x_i) \in [1, \frac{q-1}{2}]$;
4. Choose at random $\sigma_i \leftarrow Z_q$;
5. Send $\hat{x}_i = \big(\sigma_i \cdot g_1,\ (-1)^i \chi_i \sigma_i \cdot g_2,\ (-1)^i \chi_i \sigma_i^2 \cdot g_T\big)$.

Evaluation, $\mathsf{Dec}\big(\mathsf{pp},\ y_1 = (\sigma_1 + \sigma_2) g_1,\ y_2 = (\chi_1\sigma_1 - \chi_2\sigma_2) g_2,\ y_3 = (\chi_1\sigma_1^2 - \chi_2\sigma_2^2) g_T\big)$:

6. If $e(y_1, y_2) = y_3$ output 1, otherwise output 0.

Figure 1: Pairing-based computational ARE for equality

Security. Note that regardless of the inputs χ_1, χ_2, the elements y_1, y_2 that the evaluator sees are uniform and independent in G_1, G_2, respectively. The reason is that $\chi_1 \neq -\chi_2$, and therefore $\sigma_1 + \sigma_2$ and $\chi_1\sigma_1 - \chi_2\sigma_2$ are two linearly independent equations in σ_1, σ_2.

It follows that for the case $\chi_1 = \chi_2$ we get information-theoretical security: In this case the evaluator's view is just $(\rho_1 g_1, \rho_2 g_2, \rho_1\rho_2 g_T)$ (for independent uniform $\rho_1, \rho_2 \in \mathbb{Z}_q$), regardless of the actual values χ_1, χ_2.

For the case $\chi_1 \neq \chi_2$ we only get computational indistinguishability, under a hardness assumption that we call additive-squaring-DH (ASDH).

Definition 5.2 (Additive Squaring DH)**.** *The decision ASDH holds for a parameter-generator $\mathcal{G}$, if for every efficiently computable $\eta = \eta(q) \in \mathbb{Z}_q^*$, the following two distribution ensembles are computationally indistinguishable:*

$$D[\eta]_\lambda := \big(\mathsf{pp}_\lambda,\ \rho_1 g_1,\ \rho_2 g_2,\ (\rho_1^2 + \eta\rho_2^2) g_T\big) \text{ and } R_\lambda := \big(\mathsf{pp}_\lambda,\ \rho_1 g_1,\ \rho_2 g_2,\ \rho_3 g_T\big), \tag{1}$$

where $\mathsf{pp}_\lambda = (q, G_1, G_2, G_T, g_1, g_2, g_T, e) \leftarrow \mathcal{G}(1^\lambda)$ *and* $\rho_1, \rho_2, \rho_3 \leftarrow \mathbb{Z}_q$.

ASDH Implies Security. Recall that for inputs χ_1, χ_2, the evaluator view includes the public parameters pp_λ and the tuple $\big(y_1 = (\sigma_1 + \sigma_2) g_1,\ y_2 = (\chi_1\sigma_1 - \chi_2\sigma_2) g_2,\ y_3 = (\chi_1\sigma_1^2 - \chi_2\sigma_2^2) g_T\big)$. Changing variables to $\tau_1 = \sigma_1 + \sigma_2$ and $\tau_2 = \chi_1\sigma_1 - \chi_2\sigma_2$, we have $\sigma_1 = \frac{\chi_1\tau_1 + \tau_2}{\chi_1 + \chi_2}$ and $\sigma_2 = \frac{\chi_1\tau_1 - \tau_2}{\chi_1 + \chi_2}$, and

$$\begin{aligned}
\chi_1\sigma_1^2 - \chi_2\sigma_2^2 &= \chi_1(\sigma_1^2 - \sigma_2^2) + (\chi_1 - \chi_2)\sigma_2^2 \\
&= (\chi_1 + \sigma_2)(\chi_1\sigma_1 - \chi_1\sigma_2) + (\chi_1 - \chi_2)\sigma_2^2 \\
&= (\sigma_1 + \sigma_2)(\chi_1\sigma_1 - \chi_2\sigma_2) + (\sigma_1 + \sigma_2)(\chi_2 - \chi_1)\sigma_2 + (\chi_1 - \chi_2)\sigma_2^2 \\
&= \tau_1\tau_2 + \tau_1(\chi_2 - \chi_1)\sigma_2 + (\chi_1 - \chi_2)\sigma_2^2 \;=\; \tau_1\tau_2 + (\chi_1 - \chi_2)\sigma_2(\sigma_2 - \tau_1) \\
&= \tau_1\tau_2 + (\chi_1 - \chi_2)\frac{\chi_1\tau_1 - \tau_2}{\chi_1 + \chi_2}\left(\frac{\chi_1\tau_1 - \tau_2}{\chi_1 + \chi_2} - \frac{\tau_1(\chi_1 + \chi_2)}{\chi_1 + \chi_2}\right) \\
&= \tau_1\tau_2 + \frac{\chi_1 - \chi_2}{(\chi_1 + \chi_2)^2}\big(\chi_1\tau_1 - \tau_2\big)\big(-\tau_2 - \tau_1\chi_2\big) \\
&= \left(1 - \frac{(\chi_1 - \chi_2)^2}{(\chi_1 + \chi_2)^2}\right)\cdot\tau_1\tau_2 + \frac{\chi_1 - \chi_2}{(\chi_1 + \chi_2)^2}\cdot(-\chi_1\chi_2)\cdot\tau_1^2 + \frac{\chi_1 - \chi_2}{(\chi_1 + \chi_2)^2}\cdot\tau_2^2 \\
&= \underbrace{\frac{4\chi_1\chi_2}{(\chi_1 + \chi_2)^2}}_{=\alpha(\chi_1,\chi_2)}\cdot\tau_1\tau_2 \;+\; \underbrace{\frac{\chi_1\chi_2(\chi_2 - \chi_1)}{(\chi_1 + \chi_2)^2}}_{=\beta(\chi_1,\chi_2)}\cdot\tau_1^2 \;+\; \underbrace{\frac{\chi_1 - \chi_2}{(\chi_1 + \chi_2)^2}}_{=\gamma(\chi_1,\chi_2)}\cdot\tau_2^2
\end{aligned}$$

Below we denote

$$\alpha = \alpha(\chi_1, \chi_2) := \frac{4\chi_1\chi_2}{(\chi_1 + \chi_2)^2}, \quad \beta = \beta(\chi_1, \chi_2) := \frac{\chi_1\chi_2(\chi_2 - \chi_1)}{(\chi_1 + \chi_2)^2}, \quad \gamma = \gamma(\chi_1, \chi_2) := \frac{\chi_1 - \chi_2}{(\chi_1 + \chi_2)^2},$$

with all the operations in $\mathbb{Z}_q$. We note that α, β, γ are well-defined since $\chi_1 + \chi_2 \neq 0$ (and q is a prime). Also, in the case where $\chi_1 \neq \chi_2$ then α, β, γ are non-zero (since χ_1, χ_2, and $\chi_1 - \chi_2$ are all non-zero). We conclude that the evaluator's view in the case $\chi_1 \neq \chi_2$ is

$$E[\chi_1, \chi_2]_\lambda \;:=\; \big(\mathsf{pp}_\lambda,\ \tau_1 g_1,\ \tau_2 g_2,\ \big(\alpha\tau_1\tau_2 + \beta\tau_1^2 + \gamma\tau_2^2\big)g_T\big)\,,$$

where α, β, γ are non-zero scalars that depend on χ_1, χ_2, and τ_1, τ_2 are uniform in $\mathbb{Z}_q$ and independent. Also consider the following "uniform" distribution ensemble

$$U_\lambda := \left(\mathsf{pp}_\lambda,\ \tau_1 g_1,\ \tau_2 g_2,\ \tau_3 \cdot e(g_1, g_2)\right),$$

where τ_1, τ_2, τ_3 are uniform in $\mathbb{Z}_q$ and independent.

Lemma 5.3. *Assuming ASDH, for any pair of distinct $\chi_1, \chi_2 \in [1, \frac{q-1}{2}]$ the ensembles U_λ, $E[\chi_1, \chi_2]_\lambda$ are computationally indistinguishable.*

Proof. Assume towards contradiction that there exists $\chi_1, \chi_2 \in [1, \frac{q-1}{2}]$, $\chi_1, \neq \chi_2$, and a distinguisher D that can tell $E[\chi_1, \chi_2]_\lambda$ from U_λ with a non-negligible advantage $\epsilon = \epsilon(\lambda)$. We show how to use D to construct D' that breaks ASDH with the same advantage ϵ.

Let $\alpha = \alpha(\chi_1, \chi_2)$, $\beta = \beta(\chi_1, \chi_2)$, and $\gamma = \gamma(\chi_1, \chi_2)$ be the non-zero scalars as above, and set $\eta := \gamma \cdot \beta^{-1} \bmod q$ (which is well defined and non-zero since β, γ are non-zero).

The would-be distinguisher D' is given as input $(\mathsf{pp}_\lambda,\ x_1 = \tau_1 g_1,\ x_2 = \tau_2 g_2,\ x_3 = \tau_3 g_T)$, and it needs to decide if $\tau_3 = \tau_1^2 + \eta\tau_2^2$ or if τ_3 is random and independent of the other variables. Setting $y := \alpha \cdot e(x_1, x_2) + \beta \cdot x_3$, D' then executes $D(\mathsf{pp}_\lambda, x_1, x_2, y)$ and outputs whatever D does.

Analyzing the advantage of D', we express y in terms of the other variables as

$$y = \alpha \cdot e(x_1, x_2) + \beta \cdot x_3 = \underbrace{(\alpha\tau_1\tau_2 + \beta\tau_3)}_{=\tau_3'} \cdot g_T.$$

Let $\tau_3' := \alpha\tau_1\tau_2 + \beta\tau_3$. If τ_3 is random and independent of the other variables then so is τ_3', and therefore in this case D's input is drawn from U_λ. On the other hand, if $\tau_3 = \tau_1^2 + \eta\tau_2^2 = \tau_1^2 + \frac{\gamma}{\beta}\tau_2^2$ then

$$\tau_3' \;=\; \alpha\tau_1\tau_2 + \beta\tau_3 \;=\; \alpha\tau_1\tau_2 + \beta(\tau_1^2 + \frac{\gamma}{\beta}\tau_2^2) \;=\; \alpha\tau_1\tau_2 + \beta\tau_1^2 + \gamma\tau_2^2.$$

Hence, in this case the input of D' is drawn from $E[\chi_1, \chi_2]_\lambda$. It follows that the advantage of D' in distinguishing ASDH is equal to the advantage of D is distinguishing U_λ from $E[\chi_1, \chi_2]_\lambda$. $\square$

Since the evaluator's view for every $\chi_1 \neq \chi_2$ is indistinguishable from U_λ, then they are also indistinguishable from each other. Hence, we have:

Lemma 5.4. *Under ASDH, the scheme from Figure 1 is a secure ARE scheme for the equality function.*

5.1.2 ASDH Reduces to Squaring XDH

Next we argue that ASDH is indeed likely a hard problem for hard-DLOG pairing groups. We do this by showing that at least in the asymmetric case $G_1 \neq G_2$, ASDH reduces to Squaring XDH.

Lemma 5.5. *Any distinguisher for ASDH can be converted to a distinguisher for Squaring XDH with the same advantage.*

Proof. Let D be an ASDH distinguisher for some $\eta = \eta(q) \in \mathbb{Z}_q^*$ with advantage $\epsilon = \epsilon(\lambda)$. Namely, it has advantage ϵ in distinguishing $D[\eta]_\lambda$ from R_λ from Eqn. (1). We use it to construct a distinguisher D' for Squaring XDH.

D' gets as input $(\mathsf{pp}_\lambda, x_1 = \rho_1 g_1, x_2 := \rho_2 g_1)$, and it needs to decide if $\rho_2 = \rho_1^2$ or if it is a random and independent variable. To use D, D' samples uniform $\tilde{\rho} \leftarrow Z_q$. It then computes

$$\tilde{x} := \tilde{\rho} g_2, \text{ and } y := e(x_2, g_2) + \eta\tilde{\rho}^2 \cdot e(g_1, g_2) = (r_2 + \eta\tilde{r}^2) \cdot e(g_1, g_2).$$

Finally, D' executes $D(\mathsf{pp}_\lambda, x_1, \tilde{x}, y)$, and outputs whatever D does.

Clearly, if ρ_2 is random and independent of the other variables then so is $\rho_2 + \eta\tilde{\rho}^2$, and therefore in that case the input of D is distributed according to R_λ. On the other hand, if $\rho_2 = \rho_1^2$ then we have $y = (\rho_1^2 + \eta\tilde{\rho}^2) \cdot e(g_1, g_2)$, so in that case the input of D is distributed according to $D[\eta]_\lambda$. Hence, the advantage of D' in distinguishing Squaring XDH is equal to the advantage of D is distinguishing R_λ from $D[\eta]_\lambda$. $\square$

5.1.3 Aside: A Direct ARE Scheme for OT

The structure of the scheme from Figure 1 makes it easy to adjust to get directly ARE for two-party oblivious-transfer, allowing us to bypass the general transformation in Section 5.2 below. We start with an ARE scheme for Rabin-OT, and then convert it to a one-of-two string-OT using standard techniques. For our purposes, it is convenient to consider Rabin-OT as the following *randomized* function:

$$f_{rot} : \{0,1\} \times (\{0,1\} \times \{0,1\}^\ell) \to \{0,1\}^\ell, \quad f_{rot}((b,s),c) = \begin{cases} s & \text{if } b = c \\ r & \text{otherwise,} \end{cases}$$

where $r \in \{0,1\}^\ell$ is (close to) uniform and independent of s. (Note that in our context it is the evaluator that gets the result, not the party with the choice bit c.)

Recall that in the protocol from Figure 1, each party i sends three elements $(x_1^i \in G_1, x_2^i \in G_2, x_3^i \in G_3)$, and the evaluator receives $(y_1 = x_1^1 + x_1^2, y_2 = x_2^1 + x_2^2, y_3 = x_3^1 + x_3^2)$, such that $y_3 - e(y_1, y_2) = 0$ if the parties' inputs are the same, and otherwise $y_3 - e(y_1, y_2)$ is nearly uniform in G_3.

Assume for now that the input of party 2 includes $s \in G_3$ (rather than s being a bit-string). In that case, the parties can run the equality protocol from above on input bits c, b, respectively, except that Party 2 sends ${x'}_3^2 = x_3^2 + s$ instead of sending x_3^2. The evaluator will receive the same y_1, y_2 as above, but rather than y_3 it will get $y_3' = x_3^1 + {x'}_3^2 = s + y_3$. The evaluator outputs $y_3' - e(y_1, y_2)$, which is exactly what we need since:

- If $b = c$ then $y_3' - e(y_1, y_2) = y_3 + s - e(y_1, y_2) = s$;
- If $b \neq c$ then $y_3' - e(y_1, y_2) = s \pm \sigma_1 \sigma_2 g_T$, which is nearly uniform in G_T.

To send a bit-string s rather than an element in G_3, we use an "encoding" $E : G_3 \to \{0,1\}^\ell$ with the property that $E(u)$ for a uniform $u \in G_3$ yields an almost-uniform bit-string in $\{0,1\}^\ell$. Party 2 (the sender) will use the protocol above with a uniformly chosen element $u \in G_3$ in the role of the input element, and in addition will send to the evaluator the bit string $t := s \oplus E(u)$. If $b = c$ then the evaluator will get u and can therefore compute $s = E(u) \oplus t$. If $b \neq c$ then the value that the evaluator obtains will be a nearly uniform $u' \in G_3$, hence $E(u') \oplus y$ will be nearly uniform in $\{0,1\}^\ell$. Moreover, by the security proof from above, that value is indistinguishable from uniform in G_3, even conditioned on the rest of the evaluator's view.

Finally, we show the standard transformation from this string-OT scheme $\mathcal{S}_{rot}$ to 1-of-2 string OT, i.e. to compute the function

$$f_{sot}\{0,1\} \times \{0,1\}^{2\ell} \to \{0,1\}^\ell, \ f_{sot}\left(c, (s_0, s_1)\right) = s_c.$$

The parties run two instances of the Rabin-OT scheme in parallel, where:

- Party 1 uses the same choice bit c in both instances.
- Party 2 with inputs $s_0, s_1 \in \{0,1\}^\ell$ chooses a random bit $b \leftarrow \{0,1\}$ and uses input $(b, (s_b|0^\lambda))$ in the first instance and input $(1-b, (s_{1-b}|0^\lambda))$ in the second.

The evaluator decodes both instances to get s_0', s_1', and if any of them is of the form $(s|0^\lambda)$ then it outputs s. To see that this is the correct output (whp), note that:

- When $c = b = 0$ then the evaluator gets $s_0' = (s_0|0^k)$ and $s_1' = r$ (for a nearly uniform $r \in \{0,1\}^{\ell+\lambda}$), so it will output s_0 whp;
- When $c = 1, b = 0$ then the evaluator gets $s_0' = r$ and $s_1' = (s_1|0^k)$, so it will output s_1 whp;
- When $c = 0, b = 1$ then the evaluator gets $s_0' = r$ and $s_1' = (s_0|0^k)$, so it will output s_0 whp;
- When $c = b = 1$ then the evaluator gets $s_0' = (s_1|0^k)$ and $s_1' = r$, so it will output s_1 whp.

Security of this scheme follows from the security of the underlying Rabin-OT scheme, and from b being a random bit.

Remark 5.6 (A plain-model construction). *As described above, the ASDH-based construction requires a trusted setup to choose the groups and their generators. We can get a setup-free construction by settling on a less standard (but equally believable) hardness assumption, where each value of λ is deterministically mapped into some groups and generators. (For example by derandomizing the usual setup, drawing the randomness from a hash function that can be modeled as a random oracle.)*

5.2 From Equality to Any Small Function

We observe that for any boolean function f over a small domain, an ARE scheme for f can be obtained from an ARE for equality.

Lemma 5.7. *Let $f : D_1 \times D_2 \to \{0,1\}$ be a boolean function over finite domains D_1, D_2. Assume w.l.o.g. that $|D_1| \le |D_2|$, and let z be an arbitrary symbol, $z \notin D_1$. Then a secure ARE scheme $\mathcal{S}_{eq}$ for equality over the domain $D_1' = D_1 \cup \{z\}$ can be converted into a secure ARE scheme $\mathcal{S}_f$ for f, where the communication complexity of $\mathcal{S}_f$ is at most $|D_1|$ times larger than that of $\mathcal{S}_{eq}$.*

Proof. Consider the $|D_1| \times |D_2|$ truth table for f, and let $k \le |D_1|$ be (an upper bound on) the largest number of 1's in any column of this table.

The scheme $\mathcal{S}_f$. On inputs x, y, the parties run in parallel k copies of the equality scheme $\mathcal{S}_{eq}$:

- Party 1 uses their input x in all these copies.
- For Party 2, consider the column corresponding to y in the truth table, and let $x_1, x_2, \ldots, x_{k'}$ be all the possible party-1 inputs for which $f(x_i, y) = 1$. (Recall that $k' \le k$). Party 2 concatenates $k - k'$ copies of the value $z \notin D_1$, yielding a sequence of length exactly k, $(x_1, \ldots, x_{k'}, z, \ldots, z) \in {D_1'}^k$.

  Party 2 also chooses a shift amount at random $\delta \leftarrow [1, k]$. Then in the i'th copy of the equality scheme, Party 2 uses the input $x_{i-\delta}$ if $i - \delta \le k'$, and the input z if $i - \delta > k'$ (with index arithmetic modulo k).

The evaluator gets k sums $y_1, \ldots, y_k$ from the k copies of $\mathcal{S}_{eq}$, and decodes them to get the k results $b_i = \mathsf{Dec}(\mathsf{pp}, y_i) \in \{0,1\}$. It outputs 1 if there is any match $b_i = 1$, and outputs 0 if they are all 0 (i.e., no match).

Correctness. Since Party 1 uses $x \in D_1$ in all the copies of $\mathcal{S}_{eq}$ and Party 2 uses $k' \le k$ distinct inputs from D_1 and the value $z \notin D_1$, then at most one of them will be a match. Moreover, it can only be one of the x_i's for $i \le k'$ (since party 1 never inputs the value z). Thus, there is a match if and only if $x = x_i$ for some $i \le k'$, which means that $f(x, y) = f(x_i, y) = 1$.

Security. Since the underlying $\mathcal{S}_{eq}$ is a secure ARE scheme, then the transcripts of all the non-matching instances are indistinguishable from some distribution $\mathcal{D}$, whereas the matching instance (if it exists) has transcript indistinguishable from another distribution $\mathcal{D}'$. Moreover, due to the random shift amount δ, the location of the matching instance (if it exists) is random in $[1, k]$.

Therefore, for any x, y such that $f(x, y) = 0$ the evaluator's view is indistinguishable from $\mathcal{D}^k$, and for any x, y such that $f(x, y) = 1$ the evaluator's view is indistinguishable from $(\mathcal{D}^{\delta-1}, \mathcal{D}', \mathcal{D}^{k-\delta})$ for a uniform index $\delta \in [1, k]$. □

5.3 Computational ARE for General Functions

Using Lemma 5.7, we can use ARE for equality to implement ARE for Oblivious-Transfer, and then extend it to general functions making a standard use of garbled circuits (see, e.g., [AIK06]).

Lemma 5.8. *Let $f : (\{0,1\}^*)^* \to \{0,1\}^*$ be an n-party function as in Definition 3.1, and let $\{C_{\lambda,n,\ell}\}$ be a boolean circuit family that computes f. Given a secure ARE scheme $\mathcal{S}_{eq}$ for equality over domains of size three, and a secure PRG, one can construct a computationally secure ARE scheme $\mathcal{S}_f$ for f, with complexity at most $2\lambda\ell(n-1) \cdot \mathsf{complexity}(\mathcal{S}_{eq}) + O(\lambda \cdot |C_{\lambda,n,\ell}|)$.*

Proof. We build an ARE scheme for f from a garbling of $C_{\lambda,n,\ell}$ [Yao86] (which can be implemented from any secure PRG), along with the two-party ARE scheme for string-OT that we can get from Lemma 5.7.

Given the public parameters $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n, 1^\ell)$ and a description of the circuit $C_{\lambda,n,\ell}$, Party 1 will construct and send to the evaluator a garbling of $C_{\lambda,n,\ell}$, and will run with each other party ℓ instances of $\mathcal{S}_{sot}$ for strings of length λ, one for each of their input bits. In each instance, Party 1 will play the role of the sender (Party 2 from $\mathcal{S}_{sot}$) using as input the two labels for that input wire, and the other party will play the receiver (Party 1 from $\mathcal{S}_{sot}$) using the corresponding input bit as the OT choice bit. (Party 1 will also send to the evaluator the labels corresponding to its own input bits.) The evaluator will therefore receive the garbled circuit, along with one label for each input wire. It will then evaluate the garbled circuit and compute the output. Correctness and security follow from those of $\mathcal{S}_{sot}$ and the garbling scheme. For complexity, we have $\ell(n-1)$ instances of $\mathcal{S}_{sot}$, each of complexity $2\lambda \cdot \mathsf{complexity}(\mathcal{S}_{sot})$, and in addition sending the garbled circuit itself. □

Theorem 5.9 (Computational ARE from ASDH)**.** *Under the ASDH assumption, there exists a computationally secure ARE scheme for every polynomial-time computable multiparty function f.*

Proof. Follows directly from Lemmas 5.4 and 5.8, and the fact that ASDH implies a secure PRG. □

Finally, we note that by using information-theoretic garbling (cf. [IK02]), one can obtain an unconditional variant of Lemma 5.8, as stated in the following completeness theorem.

Theorem 5.10 (Equality is complete)**.** *If there is a statistical ARE for equality over domains of size three, then every function f admits a statistical ARE, with communication complexity polynomial in the branching-program size of f.*

6 Robust ARE via Obfuscation

In this section, we show that robust ARE can be implemented using the "heavy cryptographic tools" of ideal obfuscation and resettable MPC.

6.1 Resettable MPC

A resettable MPC protocol is one that remains secure even in the highly adversarial setting where the adversary can reset the honest parties to their initial state, then run them again (while sending them different messages) in the hope of extracting more information from them. This setting was initially studied in the context of zero-knowledge ([CGGM00] and follow-up), and later also in more general settings [GS09, GM11]. We assume that all messages in the protocol are transmitted over a *broadcast* channel.

We note that a resetting adversary in particular has access to the *full residual function* defined by the honest parties. Namely, it can repeatedly run the protocol with different inputs for any subset of the parties (honest or not) while keeping the inputs of the other parties fixed. The security definition therefore gives the same power also to the simulator, giving it access to this full residual function. To ensure that this is the *only* information available to an adversary who is given the obfuscated next-message functions of the resettable MPC protocol, we follow the approach of [BIK$^+$22] and use a signature-based mechanism to effect the adversary follow a valid execution path between honest parties.

Concretely, the resettable MPC protocol is augmented by adding a preamble round where each party P_i broadcasts a signature verification key vk_i, and thereafter it must sign all its outgoing messages relative to that key. Parties ignore messages that are not properly signed relative to the keys vk_i of the first round.

We sketch the definition of resettable MPC below. See [GS09, BIK$^+$22] for a more detailed version.

Definition 6.1 (Resettable-MPC). *A multi-party protocol Π is a resettably-simulatable implementation of a multi-party function f, if for any (real-world) attacker A against Π that can reset the honest parties to their initial state (with fixed input and randomness), there is an (ideal-world) simulator S controlling the same parties and with access to the full residual function, such that the output of S is indistinguishable from the view of A, even when taken jointly with the outputs of the honest parties.*

It was shown in [GM11] that not all functions have resettable MPC protocols in the plain model. On the other hand, it is a folklore result that they can be realized in the CRS model under standard cryptographic assumptions (as was mentioned in [HIJ$^+$17]). Also, in [BIK$^+$22] it was shown how to realize general resettable MPC in the plain model with super-polynomial simulation (e.g., with simulator running in time $2^{\lambda^{0.1}}$ or even in quasi-polynomial time), again under standard cryptographic assumptions.

6.2 Obfuscation

Our construction below is stated in the ideal-obfuscation model [JLLW22], in which access to the obfuscated circuit is replaced by an oracle access to the (stateless[8]) function that this circuit implements.

A protocol in the ideal obfuscation model can be made into a concrete protocol by replacing the ideal obfuscation with some iO candidate, making an ad-hoc computational assumption about the obfuscation being good enough to meet whatever notion of security we are considering. In our context, when we heuristically instantiate the ideal obfuscation, the best we can hope for is the indistinguishability variant of robust ARE. See more discussion in Remark 3.5.

[8]The oracle can use hard-wired values for state, but it cannot maintain an evolving state from one query to the next.

6.3 Constructing Robust ARE

Next we show how to construct robust ARE from resettable MPC and (ideal) obfuscation. Consider a multiparty function $f : (\{0,1\}^\ell)^n \to \{0,1\}^{\ell'}$ that we want to realize with ARE, and define the following extended multiparty function

$$\begin{aligned} g_f : \Big(\{0,1\}^\ell \times \{0,1\}^\lambda \times \{0,1\}^\lambda\Big)^n &\quad\to\quad \big(\{0,1\}^{\ell'} \cup \{\bot\}\big), \\ g_f\big((x_1,\rho_1,\sigma_1),\ldots,(x_n,\rho_n,\sigma_n)\big) &\quad=\quad \begin{cases} f(x_1,\ldots,x_n) & \text{if } \sigma_1 = \sigma_2 \cdots = \sigma_n = \bigoplus_{i\in[n]} \rho_i, \\ \bot & \text{otherwise.} \end{cases} \end{aligned} \tag{2}$$

The intention is to use the ρ's and σ's to "lock" the values of honest parties, so that the adversary can only use them to compute the residual function *with all the x_i's fixed*, and nothing more. Let Π be a resettable MPC protocol for this g_f, and denote the next-message function of Party i by

$$\mathsf{nextMsg} := M_i\big(r_i, (x_i,\rho_i,\sigma_i), \mathsf{prevMsgs}\big).$$

In the ARE scheme for f, denoted $\mathcal{S}_f$, each party P_i with input x_i chooses randomness r_i for the protocol Π and a random string ρ_i for g_f, and hard-wires r_i, x_i, ρ_i in M_i to get the function

$$M'_{i,r_i,x_i,\rho_i}\big(\sigma, \mathsf{prevMsgs}\big) := M_i\big(r_i, (x_i,\rho_i,\sigma), \mathsf{prevMsgs}\big).$$

Party P_i obfuscates the resulting function, $\hat{M}_i \leftarrow OBF(M'_{i,r_i,x_i,\rho_i})$. The encoding that it sends is $\big(\rho_i; \hat{M}_i\big)$.

Recall from Remark 3.2 that the syntax $(\rho; \hat{M})$ means that the evaluator gets the sum $\sigma := \oplus_i \rho_i$'s, and all the individual $\hat{M}_i$'s. It uses the obfuscated next-message functions to run the protocol Π, feeding it with the same σ everywhere, and as a result obtaining the value

$$g_f\big((x_1,\rho_1,\sigma),\ldots,(x_n,\rho_n,\sigma)\big) = f(x_1,\ldots,x_n).$$

6.3.1 Security Analysis

The key point is that the adversary cannot predict the sum of the ρ_i's of a strict subset of the honest parties. If the adversary uses any of the obfuscated circuits $\hat{M}_i$ of the honest parties, then it is bound to the ρ_i in that circuit. The only way for the adversary to get a matching σ is to use the one that the evaluator received. But this σ ties it also to the ρ_i's of all the other honest parties. Hence, if the adversary uses any of the obfuscated $\hat{M}_i$'s of the honest parties then it must use them all. Security then follows from security of the resettable MPC protocol and (ideal) obfuscation, which imply that query-bounded access to the obfuscated next-message functions can be simulated given access to the full residual function of g_f.

Theorem 6.2. *For any multiparty function f, if the function g_f from Eqn. (2) has a resettably simulatable protocol, then f has a simulation-robust ARE scheme in the ideal-obfuscation model.*

Proof. (sketch) Recall that in the ideal obfuscation model, the bitstrings $\hat{M}_i$ are replaced by having oracle access to all the functions $M'_{i,x_i,r_i,\rho_i}(\cdot,\cdot)$. Let A be a "real-world adversary" against the scheme $\mathcal{S}_f$ in this model, controlling a subset $B \subset [n]$ of the parties (B for Bad), and denote the set of honest parties by $H = [n] \setminus B$. The adversary comes up with ρ_j's on behalf of the parties in B, then it gets the sum $\sigma = \oplus_{i\in[n]}\rho_i$ and an oracle access to the functions $M'_{i,\dots}(\cdot,\cdot)$ of the honest parties.

We need to show a simulator S, with access to the residual function $f_{H,x}(\cdots)$, that can simulate A's view. The simulator will use A as a subroutine, alongside the resettable-MPC simulator S' for Π (that exists by Definition 6.1). Recall that S' expects to get access to the *full residual function* for g_f, that allows it to substitute the inputs to any subset of parties. On the other hand, S only has access to the (standard) residual function for f, with the x_i's fixed for all $i \in H$. To make up the difference, S chooses itself the ρ_i's for the honest parties, and we rely on the inability of A to guess the sum of any proper subset of them. After getting from A the ρ_j's for $j \in B$, S computes $\sigma = \bigoplus_{i\in[n]} \rho_i$ and returns it to A.

Later, S uses S' to simulate the access that A requires to the $M'_{i,\dots}$'s, and uses its f-residual-function oracle to implement the g_f-full-residual-function oracle that S' needs. In the g_f-full-residual function that S' sees, some of the (x_i, ρ_i, σ_i)'s of the honest parties are fixed, and others are specified in the query. For the fixed parties, the σ_i's are equal to σ that S sent to A. Below we stress the "right values of the honest parties" by denoting them with a star. Namely, the x_i^*'s are the real inputs of the honest parties (that S only has access to via the residual function f_{H,x^*}), and the ρ_i^*'s are the values chosen by S for the honest parties.

Consider now a query $\{(x_j, \rho_j, \sigma_j)\}_{j\in B'}$ that S' makes to its g_f-full-residual-function, where B' includes all the bad parties B as well as some honest parties from H.

- If $B' = [n]$ then this is just a query to g_f itself, which S can directly compute.
- If $B' = B$ then S checks that the $\sigma_j = \sigma$ for all $j \in B$, and $\bigoplus_{j\in B} \rho_j = \sigma \oplus \bigoplus_{i\in H} \rho_i^*$. If any of these checks fails then S returns $\perp$, otherwise it uses its residual function access to return $f_{H,x^*}(\{x_j\}_{j\in B})$.
- Otherwise ($B \subset B' \subset [n]$), S returns $\perp$.

In the first two cases, it is clear that S returns the right answer. For the last case, note that the answer is indeed $\perp$, unless S' (driven by A) was able to specify ρ_j's for the honest parties in $H' = H \cap B'$ such that $\sum_{j\in H'} \rho_j = \sum_{j\in H'} \rho_j^*$. Since A and S' have no information on the ρ_j^*'s beyond the sum of all of them, then this equality only holds with probability $2^{-\lambda}$. □

7 From ARE to Multiparty Randomized Encoding

The notion of a multiparty randomized encoding (MPRE) [ABT21] is a natural extension of the notion of randomized encoding of functions from [IK00, AIK06] to the multiparty setting. We say that an n-party function f has a t-secure MPRE in a function class $\mathcal{G}$ if there is an n-party function $g \in \mathcal{G}$ such that f can be realized with security against at most t *semi-honest* parties by performing a local (randomized) computation on the inputs followed by a single call to g. Here we consider an *external-output* variant of MPRE, where the output of g is public. Namely, the output of g is delivered not only to the n parties but also to an external party, who should only learn the output of f. This stronger notion of MPRE will be more convenient for our purposes. We will refer to the usual notion of MPRE as *internal-output* MPRE. Finally, unless stated otherwise, we will assume a *full security* threshold of $t = n - 1$.

The power of randomized encoding comes from the implementation class $\mathcal{G}$ being "simpler" than the original function f. Two related notions of simplicity that were studied in the literature are *algebraic degree* (say, over $\mathbb{F}_2$), and *locality.* We say that an MPRE has locality d if each output bit of g depends on the inputs of at most d *parties.*

Note that any d-local external-output MPRE can be realized via parallel calls to a $(d+1)$-party "internal-output" functionality, by just using an additional party to receive the output.

The main open question about MPRE is whether every n-party function has a degree-2 (or 2-local) information-theoretic MPRE with *full security* (i.e., with $t = n - 1$). Even in a computational setting, such a construction is only known based on (a non-black-box use of) oblivious transfer [ABT21]. The best known construction of information-theoretic degree-2 MPRE [AIKP22] is t-secure only for $t < 2n/3$.[9] Requiring the MPRE to be efficient, this holds either with perfect security for "simple" f (e.g., $f \in \mathsf{NC}^1$) or with computational security for general f, assuming one-way functions.

Insisting on full security, the following results are known.

Lemma 7.1 (Fully secure 3-local, degree-3 MPRE)**.** *[ABG$^+$20, Theorem 6.4]. Every n-party function f admits a 3-local degree-3 fully secure external-output MPRE g that consists of multiple copies of the function*

$$\mathsf{3MULTPlus}((x_1, z_1), (x_2, z_2), (x_3, z_3)) = x_1x_2x_3 + z_1 + z_2 + z_3$$

(defined over $\mathbb{F}_2$) over different sets of inputs. Requiring the MPRE to be efficient, this holds either with information-theoretic security for f in NC^1 or with computational security for general polynomial-time f, assuming one-way functions.

By Lemma 7.1, to obtain a fully-secure 2-local, degree-2 MPRE for general functions, it suffices to obtain such an MPRE for the function $\mathsf{3MULTPlus}$. We will construct such an MPRE assuming the existence of *robust* ARE for a related (but still constant-size) 3-party function. Our proof relies on the following lemma, which is a simple generalization[10] of [ABG$^+$20, Lemma 6.1].

Lemma 7.2 (2-local MPRE for sum of 2-local functions)**.** *Let g be an n-party function of the form $g(x_1, \ldots, x_n) = \sum_{1 \le i < j \le n} g_{ij}(x_i, x_j)$, where addition is over some finite Abelian group. Then, g has a perfect 2-local MPRE.*

Remark 7.3 (Public parameters vs. plain model)**.** *For simplicity, in the computational setting we will assume that the ARE works in the "plain model" without any public parameters* pp*. If the ARE does have public parameters, then so will the resulting MPRE. Note, however, that the indistinguishability variant will suffice for our purposes. Hence, we can instantiate our resettable-MPC-based construction from Theorem 6.2 with a protocol in the plain model, which (when instantiating ideal obfuscation by iO) yields a candidate for general indistinguishability-robust ARE in the plain model.*

We now prove our main technical theorem, which implies a 2-local encoding for $\mathsf{3MULTPlus}$.

Theorem 7.4 (2-local MPRE from robust ARE: The constant-size case)**.** *Suppose every 3-party function f with constant-size input domains admits an indistinguishability-robust computational ARE (resp., robust statistical ARE) in the plain model. Then, every such f admits a computationally (resp., statistically) secure 2-local (external-output) MPRE.*

[9] For standard internal-output MPRE, this can be improved to $t \le 2n/3$.

[10] The lemma from [ABG$^+$20] applies to degree-2 polynomials. Here we replace each monomial by a 2-local function.

Proof. (sketch) While the 3-party case suffices for our purposes, we will in fact prove the theorem for any n-party f with constant input size per party. Also, to simplify notation we will only consider here the statistical case. The computational case is similar.

By Lemma 7.2 and MPRE composition, it suffices to show that every function f as in the theorem statement admits an MPRE where g is a sum of 2-local functions. Since we assume that f has a robust ARE, then a first attempt is to use g that directly computes the sum of the ARE messages. (This is a degenerate special case, since each 1-local function is also 2-local.) However, even a robust ARE still allows the corrupted parties to learn the residual function induced by the inputs of the honest parties, which violates MPRE security.

To avoid leaking the residual function, we define a new N-party function f', for $N = \binom{n}{2}$ "virtual parties," which applies a simple *constant-size* pairwise multiparty authentication for the inputs of f. Concretely, for each pair of parties $1 \leq i < j \leq n$, there is a virtual party P_{ij} whose input to f' is a pair $x^{ij} = (x_i^{ij}, x_j^{ij})$. The function f' checks that all input pairs are consistent with some global input vector $(x_1, \ldots, x_n)$, outputting $f(x_1, \ldots, x_n)$ if it is and $\perp$ otherwise. Namely,

$$f'(x^{1,2}, \ldots, x^{n-1,n}) = \begin{cases} f(x_1, \ldots, x_n) & \text{if } \exists x_1, \ldots, x_n \text{ s.t. } \forall i, j, x^{i,j} = (x_i, x_j) \\ \perp & \text{otherwise} \end{cases}.$$

Note that if f has constant-size input domains then so does f'.

We now use a robust ARE for f' to define an MPRE for f in which the function g is a sum of 2-local functions. By Lemma 7.2, this suffices to get a 2-local MPRE for f. The function g takes from each party P_i the following inputs: its original input x_i, and additional inputs ρ_{ij} (for all $j \neq i$) that will be used to generate the ARE messages of virtual parties P_{ij}.

Letting $\Pi = (\mathsf{Enc}_{ij}, \mathsf{Dec})$ be a robust ARE for f', the function g is:

$$g\left(\left(x_1, (\rho_{1j})_{j \neq 1}\right), \ldots, \left(x_n, (\rho_{nj})_{j \neq n}\right)\right) = \sum_{1 \leq i < j \leq n} \mathsf{Enc}_{ij}((x_i, x_j); \rho_{ij} \oplus \rho_{ji})$$

where summation is over the ARE group, and $\mathsf{Enc}_{ij}((x_i, x_j); \rho)$ denotes an ARE encoding for f' of input (x_i, x_j) (for virtual party P_{ij}) using randomness ρ. By construction, the function g is indeed a sum of 2-local functions, as required. The output of f can be recovered from the output of g by applying the ARE decoder Dec of Π. It remains to argue that a set of corrupted parties can learn nothing (given their inputs, randomness, and the output of g) beyond the output of f.

We refer to the ARE message of virtual party P_{ij} as being *fully-corrupted* if both i, j are corrupted and *partially-honest* otherwise. We now argue that, conditioned on the adversary's inputs and randomness, the extra information revealed by the output of g can be simulated given the sum of the partially-honest ARE messages. This follows because: (1) conditioned on the adversary's randomness, a partially-honest message is distributed as it should (since either ρ_{ij} or ρ_{ji} is unknown to the adversary), and (2) g outputs the sum of all ARE messages of virtual parties, where the fully-corrupted ones can be determined by the adversary.

It remains to argue that learning the sum of the partially-honest ARE messages reveals no more than the output of f. By the robust ARE security of Π, this sum only reveals the residual function of f' restricted to partially-honest inputs (x_i, x_j). Assuming at least one party is honest, every input x_k is included in at least one such partially-honest pair. Hence, by the definition of f', the residual function induced by the partially-honest inputs depends only on $f(x_1, \ldots, x_n)$, as required. $\square$

Since a 2-local MPRE implies a degree-2 MPRE [ABG+20], Theorem 7.4 can be viewed as a barrier for ruling out a degree-2 statistical MPRE for general functions. Indeed, this would require proving the same for robust statistical ARE. Such ARE look incomparable to degree-2 statistical randomized encodings [IK00], which are another barrier for ruling out degree-2 MPRE [ABG+20]. See Section 1.3 for further discussion.

Combining Theorem 7.4 with Lemma 7.1, using a natural composition property for MPRE, we get the following.

Corollary 7.5 (MPRE from robust ARE: The general case)**.** *Suppose every constant-size 3-party function f admits an indistinguishability-robust (resp., statistically robust) ARE in the plain model. Then every n-party f admits a 2-local degree-2 (external-output) MPRE, or alternatively a non-interactive protocol using parallel invocations of a 3-party functionality.*

Requiring the MPRE to be efficient, this holds for computational security if one-way functions exist, and for statistical security if both f and the ARE encoding are in NC^1.

Proof. We obtain a 2-local degree-2 MPRE for f via MPRE composition, using the following steps. First, encode f using f' consisting of parallel copies of $\mathsf{3MULTPlus}$, as guaranteed by Lemma 7.1. Next, apply Theorem 7.4 to encode f' by a 2-local f''. This already gives a 2-local (external-output) MPRE for f, which can be evaluated using parallel calls to 3-party functionalities by using the third party to implement the external MPRE output.

Finally, we can convert any 2-local MPRE into a degree-2 MPRE without increasing the locality by (again) applying Lemma 7.1, noting that for each term $x_1x_2x_3$ appearing in an instance of $\mathsf{3MULTPlus}$ there is a party who holds at least two of the variables of the term. By multiplying these variables locally, the degree of $\mathsf{3MULTPlus}$ is reduced from 3 to 2. □

Acknowledgements. We thank Jonathan Ullman for helpful discussions on differential privacy in the shuffle model and the anonymous reviewers for their comments. Y. Ishai and E. Kushilevitz were supported by ISF grant 2774/20 and BSF grant 2018393. Y. Ishai was additionally supported by ERC Project NTSC (742754).

References

[AAP19] Navneet Agarwal, Sanat Anand, and Manoj Prabhakaran. Uncovering algebraic structures in the MPC landscape. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 381–406. Springer, 2019.

[ABG+20] Benny Applebaum, Zvika Brakerski, Sanjam Garg, Yuval Ishai, and Akshayaram Srinivasan. Separating two-round secure computation from oblivious transfer. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 71:1–71:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. Full version: `https://eprint.iacr.org/2020/116.pdf`.

[ABT21] Benny Applebaum, Zvika Brakerski, and Rotem Tsabary. Perfect secure computation in two rounds. *SIAM J. Comput.*, 50(1):68–97, 2021.

[AHI+17] Benny Applebaum, Naama Haramaty, Yuval Ishai, Eyal Kushilevitz, and Vinod Vaikuntanathan. Low-complexity cryptographic hash functions. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 7:1–7:31. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. *SIAM J. Comput.*, 36(4):845–888, 2006.

[AIK14] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. *SIAM J. Comput.*, 43(2):905–929, 2014.

[AIKP22] Benny Applebaum, Yuval Ishai, Or Karni, and Arpita Patra. Quadratic multiparty randomized encodings beyond honest majority and their applications. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 453–482. Springer, 2022.

[App17] Benny Applebaum. Garbled circuits as randomized encodings of functions: a primer. In Yehuda Lindell, editor, *Tutorials on the Foundations of Cryptography*, pages 1–44. Springer International Publishing, 2017.

[BBC+14] Boaz Barak, Nir Bitansky, Ran Canetti, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Obfuscation for evasive functions. In Yehuda Lindell, editor, *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 26–51. Springer, 2014.

[BBGN20] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. Private summation in the multi-message shuffle model. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 657–676. ACM, 2020.

[BF03] Dan Boneh and Matthew Franklin. Identity-based encryption from the weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, 2012.

[BGI+14] Amos Beimel, Ariel Gabizon, Yuval Ishai, Eyal Kushilevitz, Sigurd Meldgaard, and Anat Paskin-Cherniavsky. Non-interactive secure multiparty computation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th*

Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II, volume 8617 of *Lecture Notes in Computer Science*, pages 387–404. Springer, 2014.

[BIK+17] Kallista A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In Bhavani Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1175–1191. ACM, 2017.

[BIK+22] Saikrishna Badrinarayanan, Yuval Ishai, Dakshita Khurana, Amit Sahai, and Daniel Wichs. Refuting the dream XOR lemma via ideal obfuscation and resettable MPC. In *ITC 2022*, volume 230 of *LIPIcs*, pages 10:1–10:21, 2022.

[BOGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, page 1–10, 1988.

[CB17] Henry Corrigan-Gibbs and Dan Boneh. Prio: Private, robust, and scalable computation of aggregate statistics. In Aditya Akella and Jon Howell, editors, *14th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2017, Boston, MA, USA, March 27-29, 2017*, pages 259–282. USENIX Association, 2017.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *ACM STOC*, 1988.

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 235–244. ACM, 2000. Available from `https://ia.cr/1999/022`.

[CSU+19] Albert Cheu, Adam D. Smith, Jonathan R. Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 375–403. Springer, 2019.

[EFM+20] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Shuang Song, Kunal Talwar, and Abhradeep Thakurta. Encode, shuffle, analyze privacy revisited: Formalizations and empirical evaluation. *CoRR*, abs/2001.03618, 2020.

[FKN94] Uriel Feige, Joe Kilian, and Moni Naor. A minimal model for secure computation (extended abstract). In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 554–563. ACM, 1994.

[GM11] Vipul Goyal and Hemanta K. Maji. Stateless cryptographic protocols. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 678–687. IEEE Computer Society, 2011.

[GMPV20] Badih Ghazi, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. Private aggregation from fewer anonymous messages. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 798–827. Springer, 2020.

[GMW19] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game, or a completeness theorem for protocols with honest majority. In *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, pages 307–328. ACM, 2019.

[GS09] Vipul Goyal and Amit Sahai. Resettably secure computation. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 54–71. Springer, 2009.

[HIJ+17] Shai Halevi, Yuval Ishai, Abhishek Jain, Ilan Komargodski, Amit Sahai, and Eylon Yogev. Non-interactive multiparty computation without correlated randomness. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 181–211. Springer, 2017.

[HIKR18] Shai Halevi, Yuval Ishai, Eyal Kushilevitz, and Tal Rabin. Best possible information-theoretic mpc. In *Theory of Cryptography: 16th International Conference, TCC 2018, Part II*, page 255–281. Springer-Verlag, 2018.

[IK00] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 294–304. IEEE Computer Society, 2000.

[IK02] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan J. Eidenbenz, and Ricardo Conejo, editors, *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, volume 2380 of *Lecture Notes in Computer Science*, pages 244–256. Springer, 2002.

[IKOS06] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography from anonymity. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006)*, pages 239–248. IEEE Computer Society, 2006.

[Ish13] Yuval Ishai. Randomization techniques for secure computation. In Manoj Prabhakaran and Amit Sahai, editors, *Secure Multi-Party Computation*, volume 10 of *Cryptology and Information Security Series*, pages 222–248. IOS Press, 2013.

[JLLW22] Aayush Jain, Huijia Lin, Ji Luo, and Daniel Wichs. The pseudorandom oracle model and ideal obfuscation. *IACR Cryptol. ePrint Arch.*, page 1204, 2022.

[LLL82] A. K. Lenstra, H. W. Lenstra, and L. Lovàsz. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515—534, 1982.

[O'D14] Ryan O'Donnell. *Analysis of Boolean functions.* Cambridge University Press, 2014. Available from `https://arxiv.org/abs/2105.10386`.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *27th Annual Symposium on Foundations of Computer Science*, pages 162–167, 1986.

A Attempted Negative Results for Statistical ARE

Here we describe an (ultimately unsuccessful) attempt at extending the negative results from Section 4.2.4 from perfect to statistical privacy. We exemplify this attempt by trying to extend Corollary 4.17 to show that the two-party, three-input equality function does not have a statistical ARE scheme. That function is defined by the following truth table:

f_{eq}	1	2	3
1	1	0	0
2	0	1	0
3	0	0	1

A statistical-privacy scheme for f_{eq} consists of six input distributions over some Abelian group G (three for each party), denoted d_i, e_i $i = 1, 2, 3$, and two output distribution f_0, f_1 over the same group that are far apart ($SD(f_0, f_1) \geq 1 - \epsilon$), such that:

$$\forall\ i_1 \neq i_2 \in \{1, 2, 3\},\ SD\big(f_1,\ d_{i_1} \star e_{i_1}\big) < \epsilon \text{ and } SD\big(f_0,\ d_{i_1} \star e_{i_2}\big) < \epsilon,$$

where $\star$ denotes convolution, $SD(\cdot, \cdot)$ is the statistical distance, and ϵ is sufficiently small (say $\epsilon = 0.001$ for the example below).

Below we rule out such probability distributions (over any group), when the distance between distributions is measured using the l_2 norm. However, to get an impossibility result for statistical ARE schemes we need to show that such distributions are impossible also for the l_1 norm, and that we were not able to do.

We begin by recalling the correspondence between AREs and VMPs that was introduced in Section 4.2.3:

distr.	e_1	e_2	e_3
d_1	$d_1 \star e_1$	$d_1 \star e_2$	$d_1 \star e_3$
d_2	$d_2 \star e_1$	$d_2 \star e_2$	$d_2 \star e_3$
d_3	$d_3 \star e_1$	$d_3 \star e_2$	$d_3 \star e_3$

$\iff$

Fourier	v_1	v_2	v_3
u_1	$u_1 \odot v_1$	$u_1 \odot v_2$	$u_1 \odot v_3$
u_2	$u_2 \odot v_1$	$u_2 \odot v_2$	$u_2 \odot v_3$
u_3	$u_3 \odot v_1$	$u_3 \odot v_2$	$u_3 \odot v_3$

.

The correspondence between the two tables is a scaled discrete Fourier transform over the complex field (DFT), $\star$ denotes convolution, and $\odot$ denotes entry-wise product of complex numbers. Recall that the scaled complex DFT is a rigid transformation that maintains l_2 distances, so for all $i_1, j_1,\ i_2, j_2 \in \{1,2,3\}$ it holds that

$$\|u_{i_1} \odot v_{j_1} - u_{i_2} \odot v_{j_2}\| \;=\; \|d_{i_1} \star e_{j_1} - d_{i_2} \star e_{j_2}\|,$$

where $\|\cdot\|$ is the l_2 norm. The rest of this section is devoted to ruling out the existence of complex vectors u_i, v_j as above (in any dimension), such that the off-diagonal vectors are close to each other in l_2 norm but far from the on-diagonal vectors.

Below we say that such vectors are f-clustered (for some factor $f > 1$) if the l_2 distance between any on-diagonal and off-diagonal vectors is at least f times larger than the distances inside each of these clusters.

Definition A.1. *Fix any $n \in \mathbb{N}$, six complex vectors $u_i, v_i \in \mathbb{C}^n$, $i = 1,2,3$, and a real number $f > 1$.*

We say that the vectors u_i, v_i are f-clustered if for all $i_1, i_2, i_3, i_4, j_1, j_2, j_3, j_4 \in \{1,2,3\}$ where all the i's are different from the corresponding j's, it holds that

$$\|\underbrace{u_{i_1} \odot v_{i_1}}_{\textit{on-diag}} - \underbrace{u_{i_2} \odot v_{j_2}}_{\textit{off-diag}}\| > f \cdot \max\big(\|\underbrace{u_{i_3} \odot v_{i_3}}_{\textit{on-diag}} - \underbrace{u_{i_4} \odot v_{i_4}}_{\textit{on-diag}}\|,\ \|\underbrace{u_{i_3} \odot v_{j_3}}_{\textit{off-diag}} - \underbrace{u_{i_4} \odot v_{j_4}}_{\textit{off-diag}}\|\big).$$

To rule out distributions d_i, e_i as above relative to l_2 norm, it suffices to prove the following lemma:

Lemma A.2. *No set of complex vectors at any dimension are f-clustered for $f > \sqrt{99}$.*

Proof. Fix an arbitrary dimension n, and denote the i'th entry in a vector $v \in \mathbb{C}^n$ by $v[i]$. Fix six vectors $u_i, v_i \in \mathbb{C}^n$, $i = 1,2,3$, denote $x_{ij} := u_i \odot v_j$, let D be the set of the three on-diagonal vectors x_{ii}, and F be the set of the six off-diagonal vectors x_{ij}, $i \neq j$. Other notations that we use:

- $\Delta_D = \max_{x,y\in D} \|x-y\|^2$, $\Delta_F = \max_{x,y\in F} \|x-y\|^2$, and $\Delta = \max(\Delta_D, \Delta_F)$.
- For every index $i \in [n]$, let $\delta_{D,i} = \max_{x,y\in D} \big|x[i]-y[i]\big|^2$ and $\delta_{F,i} = \max_{x,y\in F} \big|x[i]-y[i]\big|^2$.

We first note that the δ_i's are small, they sum up to not much more than the Δ's. To see that, for any $x, y \in F$ let $A(x,y)$ be the set of indexes where $|x[i]-y[i]|^2 = \delta_{F,i}$, i.e. those indexes i where the maximum was obtained between these vectors x and y. (If the maximum was obtained in more than one pair, then choose one of them arbitrarily, we want the $A(x,y)$'s to be a partition of the index set $[n]$.) Then we have:

$$\sum_{i\in[n]} \delta_{F,i} = \sum_{x,y\in F} \sum_{i\in A(x,y)} |x[i]-y[i]|^2 \le \sum_{x,y\in F} \sum_{i\in[n]} |x[i]-y[i]|^2 = \sum_{x,y\in F} \|x-y\|^2 \le \binom{6}{2}\Delta_F = 15\Delta_F.$$

By a similar argument, we have $\sum_{i\in[n]} \delta_{D,i} \le \binom{3}{2}\Delta_D = 3\Delta_D$.

Next we show that for any two points x, y (on or off diagonal) and any index i, we have $\big|x[i]-y[i]\big|^2 \le 3\delta_{D,i} + 6\delta_{F,i}$. This holds by definition if both x, y are on-diagonal or if both are off-diagonal, so it remains to show it when x is on-diagonal and y is off-diagonal.

Fix an index i, and assume (w.l.o.g) that $|u_1[i]| \leq |u_2[i]|$ (i.e. the norm of the complex number in the i'th entry of u_1 is no larger than in u_2). For the vectors $x_{13} = u_1 \odot v_3$ and $x_{11} = u_1 \odot v_1$ we then have

$$\begin{aligned} \left|x_{13}[i] - x_{11}[i]\right|^2 &= \left|u_1[i](v_3[i] - v_1[i])\right|^2 = \left|u_1[i]\right|^2 \cdot \left|v_3[i] - v_1[i]\right|^2 \\ &\leq \left|u_2[i]\right|^2 \cdot \left|v_3[i] - v_1[i]\right|^2 = \left|x_{23}[i] - x_{21}[i]\right|^2 < \delta_{F,i}, \end{aligned}$$

(where the last inequality holds since $x_{21}, x_{23} \in F$). Then for every $x \in F, y \in D$ we get

$$\begin{aligned} \left|x[i] - y[i]\right|^2 &= \big|\underbrace{x[i] - x_{13}[i]}_{z_1} + \underbrace{x_{13}[i] - x_{11}[i]}_{z_2} + \underbrace{x_{11}[i] - y[i]}_{z_3}\big|^2 \\ &\overset{(*)}{\leq} 3(|z_1|^2 + |z_2|^2 + |z_3|^2) \leq 3(\delta_{F,i} + \delta_{F,i} + \delta_{D,i}) = 3\delta_{D,i} + 6\delta_{F,i}, \end{aligned} \tag{3}$$

as claimed. To see why inequality $(*)$ holds, note that for any two complex numbers z, z' we have $|z\bar{z'} + z'\bar{z}| \leq |z|^2 + |z'|^2$, and therefore the complex numbers z_1, z_2, z_3 above satisfy $|z_1 + z_2 + z_3|^2 = \left|\sum_{i,j} z_i \bar{z_j}\right| \leq 3 \sum_i |z_i|^2$.

Using Eqn. (3), we conclude that for any $x \in F, y \in D$

$$\|x - y\|^2 = \sum_{i \in [n]} \left|x[i] - y[i]\right|^2 \leq \sum_{i \in [n]} (3\delta_{D,i} + 6\delta_{F,i}) \leq 3 \cdot 3\Delta_D + 6 \cdot 15\Delta_F \leq 99\Delta.$$

This, in turn, means that $f^2 = \max_{x \in F, y \in D} \|x - y\|^2 / \Delta \leq 99$. □

Lemma A.2 means that if all the on-diagonal vectors are close to each other upto some ϵ in l_2 norm, and all the off-diagonal vectors are close to each other upto ϵ, then the on-diagonal cannot be more than 10ϵ away from the off-diagonal. Since DFT preserves l_2 norm, then the same holds for the distributions $d_i \star e_j$. We remark that it is possible to prove the equivalence of Lemma A.2 also for l_1 norm, but this will only prove it for the Fourier representation of the distribution, and l_1 norm is not preserved under DPT.

B Lattice-Based candidate for Computational ARE

Below we describe a candidate for computational ARE, whose security seems heuristically to be related to a "ring-LWE-like" hardness assumption. This candidate uses a similar structure to the pairing-based construction from Section 5, using the observation that the cross products in the equation $(x + y)(ax - by)$ cancel out if and only if $a = b$. Differently from the pairing construction, here we always have some cross terms due to the noise, and we attempt to make them small enough to ensure correctness.

Let λ be the security parameter, below we assume an algebraic ring of the form $R_q = \mathbb{Z}_q[X]/F(X)$ where $q > 2^{12\lambda}$ and the degree of F is $\lambda + 1$. We also assume that multiplication in R_q preserves smallness, specifically that $\|xy\| \leq \lambda^2 \cdot \|x\| \cdot \|y\|$ for all $x, y \in R_q$, where $\|\cdot\|$ is some convenient norm in some convenient representation (e.g., l_2 norm in the standard power basis). It is not hard to find rings where this condition holds. Finally, we assume input encoding such that the Euclidean norm of the inputs x_1, x_2 is always fixed (e.g., all inputs are encoded as vectors $\vec{x_i} \in \{\pm 1\}^\lambda$) and that $x_1 - x_2$ is never a zero divisor in R_q when $x_1 \neq x_2$. The scheme is described in Figure 2.

Parameters: Security parameter λ and input length ℓ.

Setup$(1^\lambda, 1^\ell)$:

1. Let $q > 2^{12\lambda}$ be an integer modulus, and $R_q = \mathbb{Z}_q[X]/F(X)$ be a ring of degree λ, where $\|xy\| \le \lambda^2 \cdot \|x\| \cdot \|y\|$ for all $x, y \in R_q$;
2. Let D_q be a distribution over R_q s.t. $s \leftarrow D_q$ yields $\|\sigma\| \le q^{2/5}$ whp;
3. Let $\mathsf{emb} : \{0,1\}^\ell \to R_q$ be injective embedding, such that $\|\mathsf{emb}(x)\| = \|\mathsf{emb}(x')\| \le q^{1/10}$ for all $x, x' \in \{0,1\}^\ell$, and $\mathsf{emb}(x) - \mathsf{emb}(x')$ is never a zero divisor in R_q when $x \ne x'$.
4. Let $a, b \leftarrow R_q$ be two random elements.

Encoding, $\mathsf{Enc}(\mathsf{pp}, i, x_i')$:

5. Embed $x_i := \mathsf{emb}(x_i')$ and choose $s_i \leftarrow D_q$;
6. Use LLL to find $e_i, f_i \in R_q$ such that $\|e_i\|, \|f_i\|, \|af_i - be_i\| < q^{2/5}$;
7. Set $u_i := s_i a + e_i$, $v_i := s_i b + f_i$;
8. Output $\big(u_i,\ (-1)^{i-1} x_i v_i,\ (-1)^{i-1} x_i u_i v_i\big)$.

Evaluation, $\mathsf{Dec}\big(\mathsf{pp},\ y = (u_1 + u_2),\ z = (x_1 v_1 - x_2 v_2),\ w = (x_1 u_1 v_1 - x_2 u_2 v_2)\big)$:

9. Output 1 if $\|yz - w\| < q^{11/12}$, 0 otherwise

Figure 2: A lattice-based computational ARE candidate

B.1 Implementing Step 2

Implementing Step 2 from Figure 2 requires solving a (structured) SIS instance. Indeed, if we let $A, B \in \mathbb{Z}_q^{\lambda\times\lambda}$ be the matrices that represent multiplications by a, b, respectively, then finding $e_i, f_i \in R_q$ as needed corresponds to finding a small integer solution to the matrix equation

$$\left[\begin{array}{c|c|c} \quad A \quad & \quad -B \quad & \quad -I \quad \end{array}\right] \times \left(\vec{e_i} \mid \vec{f_i} \mid \vec{g_i}\right)^t = \vec{0} \pmod{q}. \tag{4}$$

Above $\vec{e_i}, \vec{f_i}, \vec{g_i} \in \mathbb{Z}_q^\lambda$ are the integer vectors representing $e_i, f_i \in R_q$ and $g_i = af_i - be_i \in R_q$, respectively. Below we denote the matrix from Eqn. (4) by $M_{a,b} := \left[\ A \mid -B \mid -I\ \right] \in \mathbb{Z}_q^{\lambda\times 3\lambda}$. Luckily, the solution that we need is not too small, so we can use LLL [LLL82] to find it.[11] Specifically, consider the lattice

$$\Lambda_{a,b} := \Lambda_q^\perp(M_{ab}) = \{\vec{z} \in \mathbb{Z}^{3\lambda} : M_{a,b}\ \vec{z}^{\,t} = \vec{0} \pmod{q}\}.$$

This is a dimension-3λ integer lattice, and if q is a prime then its determinant is exactly q^λ. We can therefore use LLL to find a solution $\vec{z} \in \Lambda_{a,b}$ such that $\|\vec{z}\| \le 2^{3\lambda/4} \cdot (q^\lambda)^{1/3\lambda} = 2^{3\lambda/4+\log(q)/3}$. When $q > 2^{12\lambda}$, then we have a solution with l_2-norm bounded by $2^{3\lambda/4+\log(q)/3} < q^{1/16+1/3} < q^{2/5}$. Parsing this solution as $\vec{z} = (\vec{e_i}|\vec{f_i}|\vec{g_i})$ we have the elements $e_i, f_i \in R_q$ that we need.

B.2 Analysis

To see that correctness holds, recall that we have $\|x_i\| < q^{1/10}$ and $\|s_i\|, \|e_i\|, \|f_i\|, \|g_i\| < q^{2/5}$ (where $g_i := af_i - be_i$). The evaluator sees the values w, y, z where

$$yz \;=\; (u_1 + u_2)(x_1v_1 - x_2v_2) \;=\; \underbrace{x_1u_1v_1 - x_2u_2v_2}_{w} + \underbrace{x_1u_2v_1 - x_2u_1v_2}_{d}.$$

The evaluator sets $d := yz - w$ and outputs 1 if d is small enough. Opening up the expression for d, we have:

$$\begin{aligned} d \;=\; x_1u_2v_1 - x_2u_1v_2 \;&=\; x_1(u_2v_1 - u_1v_2) + (x_1 - x_2)u_1v_2 \\ &=\; x_1\big[(s_2a + e_2)(s_1b + f_1) - (s_1a + e_1)(s_2b + f_2)\big] + (x_1 - x_2)u_1v_2 \\ &=\; x_1\big[s_2\underbrace{(af_1 - be_1)}_{g_1} + s_1\underbrace{(af_2 - be_2)}_{g_2} + e_2f_1 - e_1f_2\big] + (x_1 - x_2)u_1v_2. \end{aligned}$$

Due to the multiplication property of R_q and the sizes of all the elements x_i, s_i, e_i, f_i, g_i, we know that in the case $x_1 = x_2$ we get

$$\begin{aligned} \|d\| \;&\le\; \|x_1s_2g_1\| + \|x_1s_1g_2\| + \|x_1e_2f_1\| + \|x_2e_1f_2\| \\ &<\; 4 \cdot \lambda^4 \cdot q^{1/10+2/5+2/5} \overset{(a)}{<} 2^{\lambda/5}q^{9/10} \overset{(b)}{<} q^{11/12}, \end{aligned}$$

where (a) holds for $\lambda \ge 156$ and (b) follows since $q > 2^{12\lambda}$. When $x_1 \ne x_2$ then we have the additional term $(x_1 - x_2)u_1v_2$, and since u_1, v_2 are pseudorandom elements then we only have a negligible probability of getting $\|d\| < q^{11/12}$.

[11] We can also get a better constants using stronger lattice reduction tools.

For security, it seems reasonable to expect the u_i's and v_i's to be pseudorandom (and pseudo-independent). These are obtained from "ring-LWE-like" expressions, except that the noise vectors are produced by LLL rather than being chosen from a "nice" distribution. If they are indeed pseudorandom in R_q, then for $x_1 \neq x_2$ the view of the adversary is just pseudorandom and pseudo-independent elements (recall that in that case $x_1 - x_2$ is invertible in R_q). For $x_1 = x_2$, since the size of x_1 is always the same then the size of what the adversary seems is not affected by the actual inputs. Hence we can hope that the distribution over d is again (pseudo)independent of the actual value of $x_1 = x_2$.