Arithmetic Cryptography

BENNY APPLEBAUM and JONATHAN AVRON, Tel Aviv University
CHRIS BRZUSKA, Technical University of Hamburg

We study the possibility of computing cryptographic primitives in a fully black-box arithmetic model over a finite field $\mathbb{F}$. In this model, the input to a cryptographic primitive (e.g., encryption scheme) is given as a sequence of field elements, the honest parties are implemented by arithmetic circuits that make only a black-box use of the underlying field, and the adversary has a full (non-black-box) access to the field. This model captures many standard information-theoretic constructions.

We prove several positive and negative results in this model for various cryptographic tasks. On the positive side, we show that, under coding-related intractability assumptions, computational primitives like commitment schemes, public-key encryption, oblivious transfer, and general secure two-party computation can be implemented in this model. On the negative side, we prove that garbled circuits, additively homomorphic encryption, and secure computation with low online complexity cannot be achieved in this model. Our results reveal a qualitative difference between the standard Boolean model and the arithmetic model, and explain, in retrospect, some of the limitations of previous constructions.

CCS Concepts: • **Theory of computation → Cryptographic primitives**; **Cryptographic protocols;**

Additional Key Words and Phrases: Arithmetic complexity, cryptography, secure computation, learning with noise

ACM Reference Format:
Benny Applebaum, Jonathan Avron, and Chris Brzuska. 2017. Arithmetic cryptography. J. ACM 64, 2, Article 10 (April 2017), 74 pages.
DOI: http://dx.doi.org/10.1145/3046675

1. INTRODUCTION

This article studies the possibility of solving cryptographic problems in a way that is independent from the underlying algebraic domain. We start by describing a concrete motivating example.

Consider the problem of computing over encrypted data where Alice wishes to store her private data $x = (x_1, \ldots, x_n)$ encrypted on a server while allowing the server to run some program f on the data. Let us assume that each data item x_i is taken from some large algebraic domain $\mathbb{F}$ (e.g., finite-precision reals) and, correspondingly, the program f is described as a sequence of arithmetic operations over $\mathbb{F}$. Naturally, Alice would like to employ a *fully homomorphic encryption* (FHE) [Gentry 2009]. However, standard FHE constructions typically assume that the data is represented as a binary string and the computation f is represented by a Boolean circuit.

A preliminary version of this article has appeared in the 6th Innovations in Theoretical Computer Science (ITCS), 2015. This work was supported by ERC starting grant 639813 ERC-CLC, ISF grant 1155/11, Israel Ministry of Science and Technology grant 3-9094, GIF grant 1152/2011, and the Check Point Institute for Information Security. The work of the third author was done while he was a postdoctoral fellow at Tel Aviv University.
Authors' addresses: B. Applebaum, School of Electrical Engineering, Tel Aviv University; J. Avron, Department of Computer Science, Tel Aviv University; C. Brzuska, Technical University of Hamburg.

DOI: http://dx.doi.org/10.1145/3046675

One way to solve the problem is to translate x and f to binary form. Unfortunately, this solution suffers from several limitations. First, such a translation is typically expensive as it introduces a large overhead (typically much larger than $\log |\mathbb{F}|$).[1] Second, such an emulation is not modular as it strongly depends on the bit-representation of x. Finally, in some scenarios, Boolean emulation is simply not feasible since the parties do not have access to the bit-wise representation of the field elements. For example, the data items $(x_1, \ldots, x_n)$ may be already "encrypted" under some algebraic scheme (e.g., given at the exponent of some group generator or represented by some graded encoding scheme [Garg et al. 2013]).

A better solution would be to have an FHE that supports $\mathbb{F}$-operations. Striving for full generality, we would like to have an FHE that treats the field or ring $\mathbb{F}$ as an oracle that can be later instantiated with any concrete domain. In this article, we explore the feasibility of such schemes. More generally, we study the following basic question:

> Which cryptographic primitives (if any) can be implemented *independently* of the underlying algebraic domain?

We formalize this question via the following notion of *arithmetic constructions* of cryptographic primitives.

1.1. The Model

Cryptographic constructions. Standard cryptographic constructions can be typically described by a tuple of efficient (randomized) algorithms P that implement the *honest* parties. The inputs to these algorithms consist of a binary string $x \in \{0, 1\}^*$ (e.g., plaintext/ciphertext) and a security parameter 1^λ that, by default, is taken to be polynomial in the length of the input x. These algorithms should satisfy some syntactic properties (e.g., "correctness") as well as some security definition. We assume that the latter is formulated via a game between an adversary and a challenger. The construction is *secure* for a class of adversaries (e.g., polynomial-size Boolean circuits) if no adversary in the class can win the game with probability larger than some predefined threshold.

Arithmetic constructions. In our arithmetic model, the input $x = (x_1, \ldots, x_n)$ to the honest parties P is a vector of generic field elements. The honest parties can manipulate field elements by applying field operations (addition, subtraction, multiplication, division, and zero-testing). There is no other way to access the field elements. In particular, the honest parties do not have access to the bit representation of the inputs or even to the size of $\mathbb{F}$. We also allow the honest parties to generate the field's constants 0 and 1, to sample random *field elements*, and to sample random *bits*. Overall, honest parties can be described by efficiently computable randomized *arithmetic circuits*. (See Section 3 for a formal definition.)

In contrast to the honest parties, the adversary is nonarithmetic and is captured, as usual, by some class of probabilistic Boolean circuits (e.g., uniform circuits of polynomial size). Security should hold for any (adversarial) realization of $\mathbb{F}$. Formally, the standard security game is augmented with an additional preliminary step in which the adversary is allowed to specify a field by sending to the challenger a Boolean circuit that implements the field operations with respect to some (adversarially chosen) binary representation. The game is continued as usual, where the adversary is now attacking the construction $P^{\mathbb{F}}$. Note that once $\mathbb{F}$ is specified, $P^{\mathbb{F}}$ can be written as a

[1] For example, for the case of finite fields with n-bit elements, the size of the best-known Boolean multiplication circuits is $\omega(n \log n)$.

standard Boolean circuit. Hence, security in the arithmetic setting guarantees that the construction $P^{\mathbb{F}}$ is secure for any concrete field oracle $\mathbb{F}$ that is realizable by our class of adversaries.[2]

Example 1.1 (*One-time Encryption*). To illustrate the model, let us define an arithmetic perfectly secure one-time encryption scheme. Syntactically, such a scheme consists of a key-generation algorithm KGen, encryption algorithm Enc, and decryption algorithm Dec that satisfy the perfect correctness condition:

$$\Pr_{k \overset{R}{\leftarrow} \mathsf{KGen}_n} [\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = m] = 1, \qquad \text{for every message } m \in \mathbb{F}^n.$$

Perfect security can be defined via the following indistinguishability game: (1) for a security parameter 1^n, the adversary specifies a field $\mathbb{F}$ and a pair of messages $m_0, m_1 \in \mathbb{F}^n$; (2) the challenger responds with a ciphertext $c = \mathsf{Enc}_k(m_b)$, where $k \overset{R}{\leftarrow} \mathsf{KGen}_n$ and $b \overset{R}{\leftarrow} \{0, 1\}$; and (3) the adversary outputs a bit b' and wins the game if $b' = b$. The scheme is perfectly secure if no (computationally unbounded) adversary can win the game with more than probability $\frac{1}{2}$.

A simple generalization of the well-known one-time pad gives rise to an arithmetic one-time encryption scheme. The key-generation algorithm samples a random key $k \overset{R}{\leftarrow} \mathbb{F}^n$; to encrypt a message $m \in \mathbb{F}^n$ we output $m+k$, and to decrypt a ciphertext $c \in \mathbb{F}^n$ we output the message $c-k$. All of these operations can be implemented by randomized arithmetic circuits. It is not hard to see that the scheme is perfectly secure. Namely, for any field $\mathbb{F}$ (or even group) chosen by a computationally unbounded adversary, the winning probability cannot exceed $\frac{1}{2}$.

1.2. Our Contribution

Our goal in this article is to find out which cryptographic primitives admit arithmetic constructions. We begin by observing that, similarly to the case of the one-time pad, typical information-theoretic constructions naturally arithmetize. Notable examples include various secret sharing schemes [Shamir 1979; Desmedt and Frankel 1991; Cramer and Fehr 2002] and classical information-theoretic secure multiparty protocols [Ben-Or et al. 1988; Chaum et al. 1988]. (See Section 1.4 for a detailed account of related works.) This raises the natural question of constructing computationally secure primitives in the arithmetic model. We give an affirmative answer to this question by providing arithmetic constructions of several computational primitives.

INFORMAL THEOREM 1.1. *There are arithmetic constructions of public-key encryption, commitment schemes, oblivious linear evaluation (the arithmetic analog of oblivious transfer), and protocols for general secure multiparty computation without honest majority (e.g., two-party computation), assuming intractability assumptions related to linear codes.*

We emphasize that our focus here is on feasibility rather than efficiency, and so we did not attempt to optimize the complexity of the constructions. The underlying intractability assumption essentially assumes the pseudorandomness of a matrix-vector

[2]Note that the computational complexity of the field representation is limited by the computational power of the adversary. Specifically, if the primitive is secure against polynomial-size circuits, then the underlying field must be implementable by a polynomial-size circuit. This limitation is inherent (for computationally secure schemes), as otherwise, one can use an inefficient field representation to break the scheme (e.g., by embedding an **NP**-complete oracle).

pair $(M, \tilde{y})$, where M is a random $m \times n$ generating matrix and $\tilde{y} \in \mathbb{F}^m$ is obtained by choosing a random codeword $y \in \mathsf{colSpan}(M)$ and replacing a random p-fraction of y's coordinates with random field elements.[3] This Random-Linear-Code assumption, which is denoted by $\mathsf{RLC}_{\mathbb{F}}(n, m, p)$, was previously put forward in Ishai et al. [2009]. If $\mathbb{F}$ is instantiated with the binary field, we get the standard *Learning Parity with Noise* (LPN) assumption [Goldreich et al. 1988; Blum et al. 1993]. Indeed, some of the primitives in this theorem can be constructed by extending various LPN-based schemes from the literature. (See Section 2.2.)

Theorem 1.1 shows that the arithmetic model is rich enough to allow highly nontrivial computational cryptography such as general secure two-party protocols. As a result, one may further hope to arithmetize *all* Boolean primitives. Our main results show that this is impossible. That is, we show that there are several cryptographic tasks that can be achieved in the standard model but cannot be implemented arithmetically. These include garbled circuits, secure computation protocols with "low" online communication, and homomorphic encryption schemes that support multiplication by a scalar or addition. Details follow.

Garbled circuits. Yao's *garbled circuit* (GC) construction [Yao 1986] maps any Boolean circuit $C : \{0, 1\}^n \to \{0, 1\}^m$ together with secret randomness into a "garbled circuit" $\hat{C}$ along with n "key" functions $K_i : \{0, 1\} \to \{0, 1\}^k$ such that, for any (unknown) input x, the garbled circuit $\hat{C}$ together with the n keys $K_x = (K_1(x_1), \ldots, K_n(x_n))$ reveal $C(x)$ but give no additional information about x. The latter requirement is formalized by requiring the existence of an efficient *decoder* that recovers $C(x)$ from $(\hat{C}, K_x)$ and an efficient *simulator* that, given $C(x)$, samples from a distribution that is computationally indistinguishable from $(\hat{C}, K_x)$. The keys are *short* in the sense that their length, k, depends only in the security parameter and does not grow with the input length or the size of C. Yao's celebrated result shows that such a transformation can be based on the existence of any pseudorandom generator [Blum and Micali 1982; Yao 1982], or equivalently a one-way function [Håstad et al. 1999].

The definition of *arithmetic garbled circuits* naturally generalizes the Boolean setting. The target function $C : \mathbb{F}^n \to \mathbb{F}^m$ is now a formal polynomial (represented by an arithmetic circuit), and we would like to *encode* it into a garbled circuit $\hat{C}$, along with n arithmetic key functions $K_i : \mathbb{F} \to \mathbb{F}^k$, such that $\hat{C}$ together with the n outputs $K_i(x_i)$ reveal $C(x)$ and no additional information about x. As in the Boolean case, we require the existence of an arithmetic decoder and simulator. We say that the garbling is *short* if the key-length depends only on the security parameter. A more relaxed notion is *online efficiency*, meaning that the key-length should be independent of the circuit complexity of C but may grow with the input length. (The latter requirement is typically viewed as part of the definition of garbling schemes; cf. Bellare et al. [2012].)

The question of garbling arithmetic circuits has been open for a long time, and only recently has some partial progress been made [Applebaum et al. 2011]. Still, so far there has been no fully arithmetic construction in which both the encoder and the decoder make a black-box use of $\mathbb{F}$. The next theorem shows that this is inherently impossible answering an open question from Ishai [2012].

[3]This is contrasted with the more standard Learning-With-Errors (LWE) assumption [Regev 2005] in which *each* coordinate of y is perturbed with some "small" element from the ring $\mathbb{Z}_p$, for example, drawn from the interval $\pm\alpha \cdot p$. Note that in the arithmetic setting, it is unclear how to sample an element from an interval that grows with p, and so LWE constructions do not seem to arithmetize. See Section 1.3 for further discussion.

INFORMAL THEOREM 1.2. *There are no short arithmetic garbled circuits. Furthermore, assuming the existence of (standard) one-way functions, even online efficient arithmetic garbled circuits do not exist.*[4]

Recall that in the Boolean setting, short garbled circuits can be constructed based on any one-way function; hence, Theorem 1.2 "separates" the arithmetic model from the Boolean model. The proof of the theorem appears in Section 5.

Secure computation with low online complexity. Generalizing the previous result, we prove a nontrivial lower bound on the online communication complexity of semihonest secure computation protocols. Roughly speaking, we allow the parties to exchange all the messages that solely depend on internal randomness at an "offline phase," and then move to an "online phase" in which the parties receive their inputs and may exchange messages based on their inputs (as well as their current view). Such an online/offline model was studied in several works [Beaver 1995; Ishai et al. 2008; Bendlin et al. 2011; Damgård et al. 2012; Ishai et al. 2013]. In the standard Boolean setting, there are protocols that achieve highly efficient online communication complexity. For example, for efficient deterministic two-party functionalities $f : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}^m$ that deliver the output to one of the parties (hereafter referred to as *simple functionalities*), one can obtain semihonest protocols with online communication of $n^{1+\varepsilon}$ based on Yao's garbled circuit, or even $n + o(n)$ based on the succinct garbled circuit of Applebaum et al. [2013]. In contrast, we show that in the arithmetic model, the online communication complexity must grow with the complexity of the function.

INFORMAL THEOREM 1.3. *Assume that arithmetic pseudorandom generators exist. Then, for every constant $c > 0$, there exists a simple arithmetic two-party functionality $f : \mathbb{F}^n \times \mathbb{F}^n \to \mathbb{F}^{n^c}$ that cannot be securely computed by an arithmetic semihonest protocol with online communication smaller than $\Omega(n^c)$ field elements.*

The existence of an arithmetic pseudorandom generator follows from the RLC assumption. The theorem generalizes to the multiparty setting including the case of honest majority. (See Section 6.)

Homomorphic encryption. A scalar-multiplicative homomorphic encryption scheme is a standard public-key encryption scheme in which, given only the public key, one can transform a ciphertext $c = \mathsf{Enc}_{\mathsf{pk}}(x)$ and a scalar $a \in \mathbb{F}$ (given in the clear) into a fresh encryption c' of the product $a \cdot x$. Formally, we require an efficient (randomized) transformation T such that, for every message $x, a \in \mathbb{F}$ and almost all public keys pk, the distributions

$$(c = \mathsf{Enc}_{\mathsf{pk}}(x), c' = T(\mathsf{pk}, c, a)) \qquad \text{and} \qquad (c = \mathsf{Enc}_{\mathsf{pk}}(x), c'' = \mathsf{Enc}_{\mathsf{pk}}(a \cdot x)) \tag{1}$$

are statistically close. We refer to this form of homomorphism as *statistical*. Two well-known examples for such schemes (over different fields) are the Goldwasser-Micali cryptosystem [Goldwasser and Micali 1984] and ElGamal cryptosystem [ElGamal 1984].

In Section 7, we show that, in the arithmetic setting, it is impossible to get scalar multiplicative homomorphism in the statistical sense. We also consider a weaker form of homomorphism, known as *multihop* [Gentry et al. 2010], which does not require Equation (1) and only asserts that correctness is preserved when T is repeatedly applied. It turns out that in this setting, it is impossible to implement arithmetic encryption

[4]The theorem holds even if the simulator is allowed to be nonarithmetic or even inefficient. The latter case corresponds to an indistinguishability notion of security.

that supports scalar multiplication and addition of ciphertexts (aka additive encryption scheme).[5]

INFORMAL THEOREM 1.4. *In the arithmetic setting, there are no encryption schemes that are (1) statistically homomorphic for scalar multiplication or (2) multihop homomorphic for addition and scalar multiplication.*

The theorem can be strengthened is several ways. For example, the first part holds even when the distributions in Equation (1) are within small constant statistical distance (e.g., 1/6), as well as in the case of one-time secure private-key encryption schemes (and, under some additional conditions, to noninteractive perfectly binding *commitments* with multiplicative homomorphism). The second part of the theorem holds even when the scheme only supports polynomially many hops or even if the scheme offers some form of "compactness" with respect to inner-product computations (see Remark 7.8).

Interestingly, we can construct, in the arithmetic setting, (one-time secure) private-key encryption schemes that support scalar multiplication (and scalar addition) with respect to a weaker form of homomorphism in which only the marginals c' and c'', defined in Equation (1), are identically distributed. Unfortunately, this form of weak homomorphism seems useless for most applications of homomorphic encryption.

1.3. Discussion and Open Questions

Taken together, our positive and negative results suggest that the arithmetic model is highly nontrivial yet significantly weaker than the standard model. Beyond the natural interest in arithmetic constructions, our negative results explain, in retrospect, some of the limitations of previous results.

For example, Applebaum et al. [2011] show that arithmetic garbled circuits can be constructed based on a special "key-shrinking" gadget, which can be viewed as a symmetric encryption over $\mathbb{F}$ with some homomorphic properties. They also provide an implementation of this gadget over the integers. This allows one to garble circuits over the ring $\mathbb{Z}_p$ in a "semiarithmetic" model, in which the encoder can treat the inputs as integers and the decoder is nonarithmetic. Theorem 1.2 shows that these limitations are inherent. Specifically, we can conclude that there are no arithmetic constructions of the key-shrinking gadget. Similarly, Theorem 1.3 partially explains the high online communication complexity of arithmetic MPC protocols such as the ones from Ben-Or et al. [1988], Chaum et al. [1988], Cramer et al. [2003], and Ishai et al. [2009].

Moreover, we believe that our results have interesting implications regarding the standard *Boolean* model. Inspired by computational complexity theory [Baker et al. 1975; Razborov and Rudich 1994; Aaronson and Wigderson 2008], one can view our negative results as some form of a barrier.

The Arithmetization Barrier: If your construction "arithmetizes," then it faces the lower bounds.

LPN/RLC versus LWE. As an example, it seems that constructions that are based on the Learning-Parity-with-Noise assumption typically extend to the arithmetic setting under the RLC assumption. Therefore, "natural" LPN-based constructions are deemed to face our lower bounds. Specifically, Theorem 1.4 suggests that it may be hard to design an LPN-based encryption with (multihop) additive homomorphism. Since such schemes can be easily constructed under Regev's Learning-With-Errors (LWE) assumption [Regev 2005], this exposes a qualitative difference between the

[5]In the conference version of this article, we stated a weaker impossibility result that applied only to restricted forms of arithmetic encryption schemes.

two assumptions. Indeed, this gap between strong LWE-type homomorphism (as in Equation (1)), which can be applied repeatedly, and weak LPN-type homomorphism, which can be applied only a small number of times, seems to be crucial. This gap may also explain why LWE has so many powerful applications (e.g., fully homomorphic encryption [Brakerski and Vaikuntanathan 2011]), while LPN is restricted to very basic primitives. The weak homomorphism supplied by typical LPN-based schemes was probably noticed by several researchers. The new insight, supplied by our arithmetic lower bound, is that the lack of multihop homomorphism is not just a limitation of a *concrete* construction but is, in fact, inherent to *all* arithmetic constructions. Quoting Pietrzak [2012], one may wonder: "... is there a fundamental reason why the more general LWE problem allows for such (rich cryptographic) objects, but LPN does not?" A simple answer would be: "LPN arithmetizes but LWE doesn't."

IT constructions. Another example for which the arithmetization barrier kicks in is the case of information-theoretic (IT) constructions. Most of the standard techniques in this domain (e.g., polynomial-based error-correcting codes) arithmetize, and so these constructions are deemed to be restricted by our lower bounds. We mention that proving lower bounds for information-theoretic primitives (even nonconstructively) is notoriously hard.[6] The arithmetic model restricts the honest parties, and as a result makes lower bounds more accessible while still capturing most existing schemes. We therefore view the arithmetic setting as a new promising starting point for proving lower bounds for information-theoretic primitives.

From a more constructive perspective, instead of thinking of arithmetic lower bounds as barriers, we may view them as road signs saying that in order to achieve some goals (e.g., basing homomorphic encryption on LPN), one must take a nonarithmetic route.

Open questions. We conclude with some open questions. First, there are several basic primitives whose existence in the arithmetic setting remains wide open. This includes Pseudorandom Functions, Collision-Resistant Hash Functions, Message Authentication Codes, and Signatures. It will also be interesting to extend our positive results to a more restricted model that does not allow one to sample random bits or to apply zero-testing. In fact, in this model we do not even know how to construct a one-way function based on a "standard assumption." On the negative side, one may ask whether our lower bounds hold in a more liberal arithmetic model in which the parties are allowed to learn an upper bound on the field size or to view a random representation of the field elements. (Such a model was considered by Ishai et al. [2009]; see Section 1.4.)

1.4. Previous Work

As already mentioned, many information-theoretic primitives admit an arithmetic implementation. Notable examples include one-time MACs based on affine functions, Shamir's secret-sharing scheme [Shamir 1979], the classical information-theoretic secure multiparty protocols of Ben-Or et al. [1988] and Chaum et al. [1988], and the randomized encodings of Ishai and Kushilevitz [2000]. Extensions of these results to generic black-box *rings* were given in Desmedt and Frankel [1991], Cramer and Fehr [2002], and Cramer et al. [2003].

Much less is known for computationally secure primitives. To the best of our knowledge, previous works only considered arithmetic models in which the honest parties

[6]A classical example is the share size of secret-sharing schemes for a general access structure. The situation becomes even more involved when it comes to more complicated objects such as secure multiparty protocols.

have a *richer* interface with the underlying field. (See below.) Therefore, the resulting constructions do not satisfy our arithmetic notion.

The IPS model. Most relevant to our work is the model suggested by Ishai et al. [2009] (hereafter referred to as the IPS model) in the context of secure multiparty computation. In this model, the parties are allowed to access the bit representation of field elements, where the field and its representation are chosen by the adversary. This allows the honest parties to learn an upper bound on the field size and to feed field elements into a standard (Boolean) cryptographic scheme (e.g., encryption, or oblivious transfer). In contrast, such operations cannot be applied in our model.[7] The work of Naor and Pinkas [1999] yields semihonest secure two-party protocols in the IPS model based on the pseudorandomness of noisy Reed-Solomon codewords. Ishai et al. [2009] extend this to the malicious model and to the case of general rings, and improve the efficiency and the underlying intractability assumptions. Both works rely on the existence of a Boolean Oblivious Transfer primitive.

Arithmetic reductions. Another line of work provides arithmetic constructions of high-level primitives P (e.g., secure computation protocol) by making use of a lower-level primitive Q (e.g., arithmetic oblivious transfer), which is defined with respect to the field $\mathbb{F}$. This can be viewed as an *arithmetic reduction* from P to Q. Arithmetic reductions from secure multiparty computation to Threshold Additive Homomorphic Encryption were given by Franklin and Haber [1993] for the semihonest model and were extended by Cramer et al. [2001] to the malicious model (assuming that the underlying encryption is equipped with special-purpose zero-knowledge protocols). Similarly, the results of Applebaum et al. [2011] can be viewed as an arithmetic reduction from garbling arithmetic circuits to the design of a special symmetric encryption over $\mathbb{F}$.

The Generic Group Model. It is instructive to compare our arithmetic model to the Generic Group Model (GGM) and its extensions [Shoup 1997; Maurer and Wolf 1998; Maurer 2005; Aggarwal and Maurer 2009]. The generic group model is an idealized model, where the adversary's computation is independent of the representation of the underlying cryptographic group (or ring). In contrast, in our model, the *honest players* are arithmetic (independent of the field), while the adversary is nonarithmetic and has the power to specify the field and its representation. Correspondingly, these two models serve very different purposes: the GGM allows one to prove unconditional hardness results against "generic attacks," while our model allows one to increase the usability of cryptographic constructions by making them "field independent." Perhaps the best way to demonstrate the difference between the models is to see what happens when the ideal oracle is instantiated with a concrete field or ring. In our model, the resulting Boolean construction will remain secure by definition, whereas in the GGM, the resulting scheme may become completely insecure [Dent 2002].

2. TECHNIQUES

2.1. Negative Results

At a high level, our main (negative) results are obtained by reducing the task of attacking arithmetic primitives to the task of "analyzing" arithmetic circuits. We solve the latter problem by making a novel use of tools (most notably partial derivatives) that were originally developed in the context of arithmetic complexity theory. Overall, our lower bounds show that *algorithms for arithmetic circuits can be used to attack*

[7]For example, in the IPS model, a party can trivially commit to a field element $x \in \mathbb{F}$ by applying a binary commitment to the bit representation of x. This is not possible in our model as x can be manipulated only via the field operations.

arithmetic constructions. Next we give an outline of the proofs of the main negative results.

For ease of presentation, we sketch (in Section 2.1.1) a version of Theorems 1.2 and 1.3 in the Private Simultaneous Messages (PSM) model of Feige et al. [1994], which is conceptually simpler than garbled circuits and general secure computation protocols. Section 2.1.2 contains an overview of the proof of Theorem 1.4.

2.1.1. Communication Lower Bounds in the PSM Model.

The PSM model. Consider two parties, Alice and Bob, who have private inputs x and y, respectively, and a shared random string r. Alice and Bob are each allowed to send a single message to a third party, Carol, from which Carol is to learn the value of $f(x, y)$ for some predefined function f, but nothing else. The goal is to minimize the communication complexity. In the standard (Boolean) setting, one can use garbled circuits to obtain a protocol in which Alice's communication depends only on her input length and the security parameter k, and is independent of Bob's input length or the complexity of f. Specifically, under standard cryptographic assumptions, Alice's message $A(x; r)$ can be of length $|x| \cdot k$ [Feige et al. 1994], or even $|x| + k$ [Applebaum et al. 2013]. In contrast, we will prove that, in the arithmetic model, the length of Alice's message $A(x; r)$ must grow with Bob's input.

Let Alice's input $x \in \mathbb{F}$ be a single field element, let Bob's input y consist of two column vectors $y_1, y_2 \in \mathbb{F}^n$, and let $f(x, (y_1, y_2)) = x \cdot y_1 + y_2$ be the target function. We will show that if Alice's message is shorter than n, Carol can learn some nontrivial information about Bob's input. In particular, Carol will output a nonzero vector that is orthogonal to y_1. (This clearly violates privacy as it allows Carol to exclude all but a $1/|\mathbb{F}|$ fraction of all possible inputs for Bob.) Let us assume, for now, that the parties do not use division or zero-testing gates, and so all the parties are simply polynomials over $\mathbb{F}$.

We begin with a few observations. Fix the shared randomness $\mathbf{r}$, Bob's input $\mathbf{y}$, and Bob's message $\mathbf{b} = B(\mathbf{y}; \mathbf{r})$, and consider the residual polynomials of Alice and Carol.[8] Alice computes a vector of univariate polynomials $A_{\mathbf{r}}(x) : \mathbb{F} \to \mathbb{F}^{n-1}$, which takes her input $x \in \mathbb{F}$ and outputs a message $a \in \mathbb{F}^{n-1}$, and Carol computes a vector of multivariate polynomials $C_{\mathbf{b}}(a) : \mathbb{F}^{n-1} \to \mathbb{F}^n$, which maps Alice's message $a \in \mathbb{F}^{n-1}$ to a vector of field elements $z \in \mathbb{F}^n$. By the correctness of the protocol, we have that

$$f_{\mathbf{y_1},\mathbf{y_2}}(x) = C_{\mathbf{b}}(A_{\mathbf{r}}(x)), \qquad \text{for every } x \in \mathbb{F}, \tag{2}$$

where $f_{\mathbf{y_1},\mathbf{y_2}}(x) = x \cdot \mathbf{y_1} + \mathbf{y_2}$. Let us fix a field $\mathbb{F}$ whose characteristic is larger than the degree of the polynomial $C_{\mathbf{b}}(A_{\mathbf{r}}(x))$.[9] Over such a large field, the univariate polynomial in the RHS of Equation (2) and the univariate polynomial in the LHS are *formally equivalent*; namely, they represent the same polynomial in $\mathbb{F}[X]$. As a result, their formal partial derivatives are also equivalent:

$$\partial f_{\mathbf{y_1},\mathbf{y_2}}(x) \equiv \partial C_{\mathbf{b}}(A_{\mathbf{r}}(x)). \tag{3}$$

By the definition of f, the LHS simplifies to $\mathbf{y_1}$, and by applying the chain rule to the RHS, we get

$$\mathbf{y_1} \equiv \mathcal{J} C_{\mathbf{b}}(A_{\mathbf{r}}(x)) \cdot \partial A_{\mathbf{r}}(x). \tag{4}$$

Syntactically, $\partial A_{\mathbf{r}}(x)$ is a (column) vector of $n-1$ univariate polynomials that contains, for each output of $A_{\mathbf{r}}(x) : \mathbb{F} \to \mathbb{F}^{n-1}$, the derivative with respect to the formal variable

[8]We use bold fonts for fixed value and standard fonts for nonfixed values, which are treated as formal variables.

[9]Since the polynomial $C_{\mathbf{b}}(A_{\mathbf{r}}(x))$ can be computed by a circuit of size $s = \text{poly}(n)$, its degree is at most 2^s and so we can just use the field $\text{GF}(p)$ where p is a prime of bit length $2s = \text{poly}(n)$.

x. Similarly, the Jacobian matrix $\mathcal{J}C_{\mathbf{b}}(a) : \mathbb{F}^{n-1} \rightarrow \mathbb{F}^{n\times n-1}$ is a matrix of multivariate polynomials whose (i, j)-th entry is the partial derivative of the ith output of $C_{\mathbf{b}}(a) : \mathbb{F}^{n-1} \rightarrow \mathbb{F}^n$ with respect to the jth input (the formal variable a_j).

Let us now get back to Carol's attack. Carol does not know $\mathbf{r}$ and therefore she cannot compute either $A_{\mathbf{r}}(x)$ or its derivative $\partial A_{\mathbf{r}}(x)$. However, she knows $\mathbf{b}$ and therefore can compute a circuit for $C_{\mathbf{b}}$, which, by using standard techniques, can be transformed into a circuit for the Jacobian $\mathcal{J}C_{\mathbf{b}}$. Carol also received from Alice a message $\mathbf{a} = A_{\mathbf{r}}(\mathbf{x})$, where $\mathbf{x}$ is Alice's input, and so Carol can evaluate the circuit $\mathcal{J}C_{\mathbf{b}}$ at the point $\mathbf{a}$ and obtain the matrix $\mathbf{M} = \mathcal{J}C_{\mathbf{b}}(\mathbf{a}) \in \mathbb{F}^{n\times(n-1)}$. Now, the key observation is that

$$\mathbf{y_1} = \mathbf{M} \cdot \mathbf{v}, \qquad \text{for some (unknown) vector } \mathbf{v}.$$

Indeed, this follows by evaluating the RHS of Equation (4) at the point $\mathbf{x}$ (and taking $\mathbf{v} = \partial A_{\mathbf{r}}(\mathbf{x})$). Overall, Carol now holds a matrix $\mathbf{M}$ whose columns span Bob's input $\mathbf{y_1} \in \mathbb{F}^n$. Since $\mathbf{M}$ has only $n - 1$ columns, Carol can find a nonzero vector that is orthogonal to $\mathbf{y_1}$ and break the security of the protocol.

Handling zero-test gates. If the parties use zero-test gates, then the functions computed by Alice and Carol are not polynomials anymore. As a result, Equation (3) does not hold since the partial derivative of the function $P(x) = C_{\mathbf{b}}(A_{\mathbf{r}}(x))$ is not defined. To solve the problem, we show that it is possible to remove the zero-test gates. Assume, for simplicity, that the circuit $P(x)$ contains a single zero-test gate that is applied to the expression $Q(x)$. Note that $Q(x)$ is a polynomial of degree d that is much smaller than the field. We distinguish between two cases: if Q is the zero polynomial, we remove the gate and replace its outcome with the constant 0; otherwise, we replace the gate with the constant 1. This transformation changes the value of P on at most d points (the roots of Q), and therefore, the resulting polynomial P' agrees with the polynomial $f_{\mathbf{y_1},\mathbf{y_2}}$ on all but d points. Since both functions are low-degree polynomials, we conclude that they must be equal. This argument easily generalizes to a large number of zero-test gates.

Some technicality arises due to the fact that the attacker Carol does not have access to P and can only compute its "outer part" $C_{\mathbf{b}}$. To see the problem, imagine that $C_{\mathbf{b}}$ contains a zero-check gate that is applied to a nonzero polynomial Q that vanishes over the image of $A_{\mathbf{r}}$. In this case, the previous procedure (applied to $C_{\mathbf{b}}$ alone) will fail miserably. We solve this issue by showing that, given a random point in the image of $A_{\mathbf{r}}$, one can remove the zero-test gates from $C_{\mathbf{b}}$ in a way that is consistent with the "inner part" $A_{\mathbf{r}}$. Since Carol can get such a point $a = A_{\mathbf{r}}(x)$ from Alice, the attack goes through. The more general setting in which the parties may also use division gates is handled similarly (except for some minor technicalities).

Extensions. The previous argument shows that Alice's communication grows with the length of Bob's input. A stronger result would say that Alice's communication grows with the complexity of the function (even if Bob's input is also short). We can prove such a result via the use of a pseudorandom generator (PRG). Roughly speaking, we embed a PRG in the function f such that a low communication protocol allows us to break the pseudorandomness of the PRG. This approach extends to the setting of arithmetic garbled circuits and general secure multiparty protocols yielding Theorems 1.2 and 1.3.

2.1.2. Impossibility of Homomorphic Encryption. Theorem 1.4 strongly relies on the existence of an efficient algorithm for the following promise problem, denoted *Arithmetic Predictability* (AP): Given a pair of arithmetic circuits $P : \mathbb{F}^n \rightarrow \mathbb{F}^m$ and $T : \mathbb{F}^n \rightarrow \mathbb{F}$, distinguish between the case where:

—(Predictable) For a randomly chosen $\boldsymbol{x} \xleftarrow{R} \mathbb{F}^n$, the random variable $T(\boldsymbol{x})$ is predictable given $P(\boldsymbol{x})$; that is, there exists an efficient[10] predictor that given $P(\boldsymbol{x})$ can guess, with high probability, the value of $T(\boldsymbol{x})$; and

—(Unpredictable) For a randomly chosen $\boldsymbol{x} \xleftarrow{R} \mathbb{F}^n$, the random variable $T(\boldsymbol{x})$ is (information-theoretically) unpredictable given $P(\boldsymbol{x})$; that is, any (computationally unbounded) predictor that gets to see $P(\boldsymbol{x})$ fails, with high probability, to guess the value of $T(\boldsymbol{x})$.

To prove Theorem 1.4, we show that attacking homomorphic encryption reduces to solving AP. We focus here, for simplicity, on the case of scalar-multiplicative statistical homomorphism (the first part of Theorem 1.4). Given a public-key pk and a ciphertext $c = \mathsf{Enc}_{\mathsf{pk}}(b)$ of an unknown plaintext $b \in \{0, 1\}$, we use the multiplicative homomorphism to construct the circuit $P(a) = f_{c,\mathsf{pk}}(a)$ that maps a plaintext $a \in \mathbb{F}$ into a fresh encryption of $a \cdot b$. Consider the probability distribution of $P(\boldsymbol{a})$ induced by a uniform choice of $\boldsymbol{a} \xleftarrow{R} \mathbb{F}$ and the internal randomness of the homomorphic evaluator (here c and pk are viewed as fixed constants). If c is an encryption of 0, then $P(\boldsymbol{a})$ is simply a fresh encryption of the zero element, and P loses all information regarding $\boldsymbol{a}$. As a result, $\boldsymbol{a}$ is unpredictable given $P(\boldsymbol{a})$. In contrast, if c is an encryption of 1, then $P(\boldsymbol{a})$ is a fresh encryption of $\boldsymbol{a}$, and so $\boldsymbol{a}$ can be predicted given $P(\boldsymbol{a})$ (e.g., by using the decryption algorithm). Hence, an efficient algorithm for predictability allows us to break the security of the multiplicative homomorphic encryption.

Building on the techniques of Dvir et al. [2009], we design an algorithm that solves AP in the case where the underlying field $\mathbb{F}$ is sufficiently large and the circuits (P, T) compute *polynomials* (i.e., do not use division gates, zero-testing gates, and random bits). In fact, the algorithm in this case is surprisingly simple: choose a random point $\boldsymbol{x} \xleftarrow{R} \mathbb{F}^n$ and check if the rows of the Jacobian $\mathcal{J}P(\boldsymbol{x}) \in \mathbb{F}^{m\times n}$ span the gradient $\partial T(\boldsymbol{x}) \in \mathbb{F}^n$ of the target polynomial $T(x)$. We further show that the case of a generalized arithmetic circuit P with division gates, zero-testing gates, and random bits reduces to the case where the circuit $P : \mathbb{F}^n \to \mathbb{F}^m$ computes polynomials. (The reduction introduces some "error terms," which force us to consider a more robust form of AP. Fortunately, the previous algorithm generalizes to this setting as well. See Section 7.)

2.2. Positive Results

Our positive results (Theorem 1.1) are based on three different approaches, outlined next.

2.2.1. Arithmetic/Binary Symmetric Encryption. One main approach is based on a new abstract notion of *arithmetic/binary symmetric encryption* (ABE). An ABE is an arithmetic symmetric encryption scheme that allows one to encrypt a field element using a binary key. That is, while the scheme works in the arithmetic model, the key is essentially a string of bits given as a sequence of 0-1 field elements. Such an encryption scheme allows us to import binary constructions to the arithmetic setting and can therefore be viewed as a bridge between the binary world and the arithmetic world.

Given, for example, a standard *binary* public-key encryption scheme, we obtain a new *arithmetic* public-key encryption by working in a hybrid mode. Namely, to encrypt a message $x \in \mathbb{F}$, encrypt x via the ABE under a fresh private binary key k, and then

[10]In fact, only the degree of P should be upper-bounded (by $2^{\mathrm{poly}(n)}$), while its circuit size may be arbitrary.

use the binary public-key encryption to encrypt the binary message k. Conveniently, for this purpose it suffices to have a *one-time* secure ABE.[11]

Similarly to the case of public-key encryption, ABE can be used to obtain arithmetic constructions of CPA-secure symmetric-key encryption and commitment schemes. In order to achieve arithmetic secure computation protocols, we will need an additional "weak homomorphism property": given a ciphertext $E_k(x)$ and field elements $a, b \in \mathbb{F}$, it should be possible to generate a new ciphertext c' that decrypts to $ax + b$. (The new ciphertext c' does not have to look like a fresh ciphertext—hence the term "weak homomorphism"—and so this does not contradict our negative results.) For technical reasons, we also require a "simple" decryption algorithm (e.g., one that can be implemented by a polynomial-size arithmetic formula or branching program).

ABE based on RLC. We show that such a one-time secure ABE can be obtained under the (generalized) Random Linear Code assumption $\mathsf{RLC}_{\mathbb{F}}(n, m, p)$. To encrypt a message x, sample a random generating matrix $A \overset{R}{\leftarrow} \mathbb{F}^{m\times n}$ together with a random p-noisy codeword y, encode the message x via a repetition code, and use the noisy codeword y to mask the encoded message $x \cdot \mathbf{1}_m$. The resulting ciphertext consists of the pair $(A, y + x \cdot \mathbf{1}_m)$. The private key is the set of all noisy coordinates, described as a binary vector. Decryption can be implemented by ignoring the noisy coordinates and solving a set of linear equations over $\mathbb{F}$. For properly chosen constants m/n and p, the system will have a unique solution, with all but negligible probability.

From ABE to secure computation. Let us explain how to construct secure arithmetic two-party computation from an ABE with weak homomorphism. The construction can be viewed as a variant of the construction of Ishai et al. [2009]. Recall that a (binary) one-out-of-two oblivious transfer ($\binom{2}{1}$-OT) is a two-party functionality that takes two inputs $a_0, a_1 \in \{0, 1\}^n$ from a sender and a selection bit $x \in \{0, 1\}$ from the receiver and delivers to the receiver the value a_x.

We begin by converting a maliciously secure binary $\binom{2}{1}$-OT into a maliciously secure $\binom{2}{1}$ *arithmetic* oblivious transfer ($\binom{2}{1}$-AOT) in which the sender's inputs $a_0, a_1 \in \mathbb{F}$ are two field elements. The transformation uses an ABE in the natural way: the sender encrypts the arithmetic messages a_0 and a_1 under binary keys k_0, k_1 and sends the ciphertexts to the receiver; then the receiver uses the binary $\binom{2}{1}$-OT to select one of two keys k_0, k_1.

Next, we convert $\binom{2}{1}$-AOT to Oblivious Linear Evaluation (OLE). The latter functionality takes two field elements $a, b \in \mathbb{F}$ from a sender and another field element $x \in \mathbb{F}$ from the receiver and delivers to the receiver the value $ax + b$. The construction makes use of the ABE again, this time exploiting the weak homomorphism. Specifically, the receiver sends the ciphertext $c = E_k(x)$, and the sender uses the homomorphism to generate a ciphertext c', which decrypts to $ax + b$. This ciphertext cannot be sent back to the receiver as it leaks information on a and b. Instead, a secure two-party computation protocol for decrypting c' is invoked. Since the input of the receiver is binary (and decryption has low complexity), such a protocol can be implemented efficiently via a $\binom{2}{1}$-AOT (e.g., via the protocols of Cramer et al. [2003]).[12] This gives a semihonest OLE.

At this point, we can use the semihonest OLE together with an arithmetic variant of the classical GMW protocol [Goldreich et al. 1987] to obtain an arithmetic secure

[11] Although only one-time security is required, ABE cannot be achieved unconditionally as the message space (the size of $\mathbb{F}$) is larger than the key space, which depends only on the security parameter and cannot grow with $\mathbb{F}$.

[12] For our concrete ABE, one can directly use $\binom{2}{1}$-AOT to securely deliver a rerandomized version of c'.

computation protocol for general arithmetic functions in the semihonest model. This protocol can be transformed to the malicious setting using the IPS compiler [Ishai et al. 2008]. To make the compiler work in our arithmetic setting, we need two additional tools: arithmetic multiparty protocol with security against a constant fraction of malicious parties (which can be constructed based on Ben-Or et al. [1988] or Cramer et al. [2003]) and a maliciously secure $\binom{2}{1}$-AOT (which we already constructed).

2.2.2. Alternative Approaches. Let us briefly mention two alternative approaches that can be used to derive arithmetic constructions for some of the primitives mentioned in Theorem 1.1.

Arithmetizing LPN-based scheme. As already mentioned, existing LPN-based schemes easily extend to the arithmetic setting under the (generalized) Random Linear Code assumption. This gives alternative arithmetic constructions for primitives like symmetric encryption [Gilbert et al. 2008], commitments [Applebaum et al. 2010; Kiltz et al. 2011], and even public-key encryption [Alekhnovich 2003] and $\binom{2}{1}$-AOT. This "direct approach" is inferior to the first (ABE-based) approach in terms of the strength of the underlying assumption. For example, using the direct approach, in order to obtain an arithmetic public-key encryption, we have to assume $\mathsf{RLC}(n, m, p)$ for constant-rate code $m = O(n)$ and subconstant noise rate $p = O(1/\sqrt{n})$. In the case of CPA-secure symmetric encryption, the direct approach requires hardness for *any* polynomial $m = m(n)$ and constant noise p. In contrast, for both primitives, the ABE-based approach requires only hardness for constant rate codes $m = O(n)$ and constant noise rate p. While all three assumptions are consistent with our knowledge, the third assumption is formally weaker than (i.e., implied by) the first two. Nevertheless, we also provide proofs based on the direct approach, as it is beneficial to demonstrate that known LPN-based schemes generalize to the arithmetic setting. (See discussion in Section 1.3.)

Arithmetizing cryptographic transformations. Another way to construct arithmetic primitives is to start with some concrete construction of a simple primitive P and then use a standard (binary) cryptographic transformation from P to a more complex primitive Q. For this, we have to translate the binary transformation to the arithmetic setting. Indeed, in some cases, existing binary transformations have a straightforward arithmetic analog. For example, we already mentioned that the classical GMW construction [Goldreich et al. 1987] of semihonest secure computation from oblivious transfer (OT) naturally extends to the arithmetic setting [Ishai et al. 2009]. Similarly, we show that Naor's transform from PRGs to commitments has an arithmetic analog. This provides another arithmetic construction of commitments whose security can be reduced to the RLC assumption.

Interestingly, some binary cryptographic transforms do not seem to arithmetize. This typically happens if the construction inspects some input $x_i \in \{0, 1\}$ and applies different operations depending on whether x_i equals zero or x_i equals one. This kind of arbitrary conditioning cannot be implemented in the arithmetic setting as x_i varies over a huge (possibly exponential size) domain. As a typical example, consider the classical GGM construction [Goldreich et al. 1986] of pseudorandom functions (PRFs) from PRGs. In the GGM construction, the value of the PRF F_k on a point $x \in \{0, 1\}^n$ is computed by walking on an exponential size tree of length-doubling PRGs, where the ith step is chosen based on the ith bit of the input. It is not clear how to meaningfully adopt such a walk to the arithmetic case in which $x_i \in \mathbb{F}$. Similar "conditioning structure" appears in the Goldreich-Levin construction of hardcore predicates [Goldreich and Levin 1989] and Yao's construction of garbled circuits from one-way functions. In fact, in the latter case, our negative results show that finding an arithmetic analog of the binary construction is *provably impossible*. The problem of proving a similar negative result for the case of PRF or, better yet, coming up with an arithmetic construction of a PRF is left open for future research.

Organization. Following some preliminaries (Section 3) and definitions of arithmetic cryptographic primitives (Section 4), the main body of this work is divided into two parts. Part 4.5.2 is dedicated to lower bounds and includes a section for each primitive: Garbled Circuits (Section 5), Secure Computation (Section 6), and Homomorphic Encryption (Section 7). The positive results appear in Part 7.5.4, beginning with a presentation of the Random Linear Code intractability assumption (Section 8) and proceeding with a section for each primitive: Pseudorandom Generators (Section 9), Encryption Schemes (Section 10), Commitments (Section 11), and Secure Computation Protocols (Section 12).

3. PRELIMINARIES

In this section, we provide some general preliminaries. We begin with standard background on probabilistic notations such as indistinguishability, entropy, and hashing (Section 3.1) and basic facts about polynomials, rational functions, and their derivatives (Section 3.2), and proceed with somewhat nonstandard definitions of efficient field representations (Section 3.3) and generalized arithmetic circuits (Section 3.4).

3.1. Probabilistic Notions

Statistical distance and indistinguishability. A real-valued function $\mu(n)$ is *negligible* if for any constant $c > 0$, $\mu(n) < n^{-c}$ for all sufficiently large ns. We use $\mathrm{neg}(\cdot)$ to denote an unspecified negligible function. We say that a sequence of events E_n holds with *overwhelming* probability if $\Pr[E_n] > 1 - \mathrm{neg}(n)$. Let P and Q be two distributions over the finite domain $\mathcal{U}$; we denote the *statistical distance* between P and Q by $\Delta(P, Q) := \frac{1}{2}\sum_{x\in\mathcal{U}} |\Pr P(x) - \Pr Q(x)|$. When $\Delta(P, Q) = 0$, we say the distributions are *identically distributed* and denote this by $P \overset{i}{\equiv} Q$. We say that two distribution ensembles $\{P_n\}$ and $\{Q_n\}$ are *statistically indistinguishable* (denoted by $P_n \overset{s}{\approx} Q_n$) if $\Delta(P_n, Q_n) = \mathrm{neg}(n)$. The ensembles $\{P_n\}$ and $\{Q_n\}$ are *computationally indistinguishable* (denoted by $P_n \overset{c}{\approx} Q_n$) if for every polynomial-size family of circuits[13] $\mathcal{A} = \{\mathcal{A}_n\}$, it holds that

$$\left| \Pr_{x \overset{R}{\leftarrow} P_n} [\mathcal{A}_n(1^n, x) = 1] - \Pr_{x \overset{R}{\leftarrow} Q_n} [\mathcal{A}_n(1^n, x) = 1] \right| = \mathrm{neg}(n).$$

Entropy. For $p \in (0, 1)$ and an integer $q > 1$, we denote by $H_q(p)$ the q-ary entropy function defined as $H_q(p) := -p\log_q(p) - (1-p)\log_q(1-p)$. The *min-entropy* $\mathbf{H}_\infty(X)$ of a random variable X distributed over a finite domain is defined as $\min_{x\in\mathrm{Supp}(X)} \log(\frac{1}{\Pr[X=x]})$. For jointly distributed random variables (X, Y), we define the *predictability* [Dodis et al. 2008] of Y given X by $\mathbf{Pred}(Y|X) = \max_{\mathcal{A}} \Pr[\mathcal{A}(X) = Y]$, where the maximum ranges over all possible (inefficient) algorithms $\mathcal{A}$. It is not hard to verify that the best guessing strategy for $\mathcal{A}(x)$ is to output the heaviest element y in the conditional distribution $(Y|X = x)$; hence,

$$\mathbf{Pred}(Y|X) \overset{\mathrm{def}}{=} \mathop{\mathbb{E}}_{x \overset{R}{\leftarrow} X}\left[\max_y \Pr[Y = y|X = x]\right] = \mathop{\mathbb{E}}_{x \overset{R}{\leftarrow} X}\left[2^{-\mathbf{H}_\infty(Y|X=x)}\right].$$

[13]We could use a uniform variant of indistinguishability; however, for our positive results (especially for secure computation), the nonuniform version is more natural and simplifies the treatment of auxiliary inputs. (The latter are necessary for composition theorems; cf. [Goldreich 2004, Chapter 7].)

A logarithmic version of predictability is captured by *Average Min-Entropy* [Dodis et al. 2008]:

$$\tilde{\mathbf{H}}_\infty(Y|X) \stackrel{\text{def}}{=} -\log(\mathbf{Pred}(Y|X)).$$

We will need the following useful facts, which include (1) a variant of Markov's inequality for average min-entropy, (2) the fact that applying a function to a random variable can only lose information, and (3) that conditioning on a random variable with λ bits of output can only reduce the average min-entropy by λ.

FACT 3.1. *Let W, X, Y be (possibly correlated) random variables. Then:*

(*1*) *For any $\delta > 0$, it holds that*

$$\Pr_{w \xleftarrow{R} W}\left[\tilde{\mathbf{H}}_\infty(Y_{W=w}|X_{W=w}) \geq \tilde{\mathbf{H}}_\infty(Y|X, W) - \log(1/\delta)\right] \geq 1 - \delta,$$

where $Y_{W=w}, X_{W=w}$ denote the joint distribution of (X, Y) conditioned on the event $W = w$. In particular, $\Pr_{x \xleftarrow{R} X}[\mathbf{H}_\infty(Y|X = x) > \tilde{\mathbf{H}}_\infty(Y|X) - \log(1/\delta)] > 1 - \delta$.

(*2*) *For every function g, it holds that $\tilde{\mathbf{H}}_\infty(Y|g(X)) \geq \tilde{\mathbf{H}}_\infty(Y|X)$. Furthermore, this holds even if g is a randomized function whose internal random coins are statistically independent of Y.*

(*3*) *If W has at most 2^λ possible values, then*

$$\tilde{\mathbf{H}}_\infty(Y|W, X) \geq \tilde{\mathbf{H}}_\infty((Y, W)|X) - \lambda \geq \tilde{\mathbf{H}}_\infty(Y|X) - \lambda.$$

PROOF. (1) By definition,

$$2^{-\tilde{\mathbf{H}}_\infty(Y|(X,W))} = \mathop{\mathbb{E}}_{w \xleftarrow{R} W}\left[2^{-\mathbf{H}_\infty(Y_{W=w}|X_{W=w})}\right];$$

hence, by Markov's inequality,

$$\Pr_{w \xleftarrow{R} W}\left[2^{-\mathbf{H}_\infty(Y_{W=w}|X_{W=w})} \geq 2^{-\tilde{\mathbf{H}}_\infty(Y|(X,W))}/\delta\right] \leq \delta,$$

and the first item follows by taking logarithms.

To prove the second part, note that if $\mathcal{A}$ predicts Y given $g(X)$ with probability p, then we can define $\mathcal{A}'$, which predicts Y given X with probability $p' \geq p$ by letting $\mathcal{A}(x) = \mathcal{A}(g(x))$. Hence, $\mathbf{Pred}(Y|g(X)) \leq \mathbf{Pred}(Y|X)$. Finally, the third item is proved in Dodis et al. [2008, Lemma 2.2.b]. □

Hashing. A family of keyed functions $\{h_k : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ are *pairwise independent hash functions* if for every $x \neq x' \in \mathcal{X}$ and $y, y' \in \mathcal{Y}$, it holds that $\Pr_{K \xleftarrow{R} \mathcal{K}}[h_K(x) = y \wedge h_K(x') = y'] = 1/|\mathcal{Y}|^2$. The following generalization of the well-known leftover hashing lemma [Håstad et al. 1999] shows that any family of pairwise independent hash functions can extract randomness from sources with high average min-entropy:

FACT 3.2 (LEMMA 2.4 OF DODIS ET AL. [2008]). *Let $\{h_k : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ be a family of pairwise independent hash functions, and let X, I be jointly distributed random variables where X is distributed over $\mathcal{X}$ and $\log|\mathcal{Y}| \leq \tilde{\mathbf{H}}_\infty(X|I) - 2\log(1/\varepsilon) + 2$. Then,*

$$\Delta((K, I, h_K(X)), (K, I, Y)) \leq \varepsilon,$$

where $K \xleftarrow{R} \mathcal{K}$ and $Y \xleftarrow{R} \mathcal{Y}$.

3.2. Polynomials and Rational Functions

Notation. We let $\mathbb{F}$ denote a finite field. For a vector $\boldsymbol{x} \in \mathbb{F}^n$, we use the notation $|\boldsymbol{x}|$ to denote the number of elements in the vector $\boldsymbol{x}$. By $w(\boldsymbol{x})$, we denote the number of nonzero elements in $\boldsymbol{x}$. For a vector in $\mathbb{F}^n$ where all coordinates have the same value $\boldsymbol{x}$, we use the notation $\boldsymbol{x}^n$. We denote the inner product of two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ by $\boldsymbol{x} \cdot \boldsymbol{y} := \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{y}_i$. If $\boldsymbol{x} \in \mathbb{F}$ is a scalar and $\boldsymbol{y} \in \mathbb{F}^n$ is a vector, then $\boldsymbol{x} \cdot \boldsymbol{y}$ stands for scalar multiplication.

We will use bold fonts to emphasize the distinction between a formal variable x and its assignment on a point $\boldsymbol{x} \in \mathbb{F}$. A multivariate monomial $M(x)$ in variables $x = (x_1, \ldots, x_n)$ over a finite field $\mathbb{F}$ is defined as $M(x) = ax_1^{c_1} \cdots x_n^{c_n}$, where $a \in \mathbb{F}$ and c_i are positive integers. A multivariate polynomial $P(x)$ is a sum of monomials. Any formal polynomial $P(x_1, \ldots, x_n)$ naturally induces a function $P : \mathbb{F}^n \to \mathbb{F}$. A pair of polynomials $P(x)$ and $Q(x)$ are *formally equivalent* (denoted by $P \equiv Q$) if they compute the same formal polynomial; that is, each monomial $M(x)$ appears in P and Q with the same coefficient. Clearly, if $P \equiv Q$, then the corresponding functions are also equal. The converse direction also holds as long as the degrees are smaller than the characteristic of the field. We will often use the following standard upper bound on the number of roots of a degree d polynomial.

LEMMA 3.3 (SCHWARTZ [1980] AND ZIPPEL [1979]). *Let $f(x_1, \ldots, x_n)$ be a nonzero polynomial of degree at most d over a field $\mathbb{F}$; then the number of roots of f is at most $d \cdot |\mathbb{F}|^{n-1}$ and therefore the probability that $f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = 0$ for random $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ is smaller than $\frac{d}{|\mathbb{F}|}$.*

A rational function f is a quotient $v(x)/u(x)$ of two polynomials $v(x)$ and $u(x)$, where $u(x)$ is not the zero polynomial. The degree of a rational function is the maximum of the degrees of its constituent polynomials v and u. Note that f is a partial function that is undefined at points $\boldsymbol{x}$, where $u(\boldsymbol{x}) = 0$. We say that a pair of rational functions $f(x) = f_1(x)/f_2(x)$ and $g(x) = g_1(x)/g_2(x)$ are *equal* (denoted $f = g$) if they agree on all inputs for which they are both defined. The functions f and g are *formally equivalent* (denoted by $f \equiv g$) if the polynomials $f_1(x)g_2(x)$ and $f_2(x)g_1(x)$ are formally equivalent. Clearly, if $f \equiv g$, then the corresponding functions are also equal. The converse direction also holds as long as the degrees are smaller than $|\mathbb{F}|/3$ (as follows by a simple application of the Schwartz-Zippel lemma).

Derivatives. We proceed with standard definitions of formal derivatives of multivariate polynomials and rational functions over finite fields together with some basic useful facts. (For a comprehensive treatment of derivatives over finite fields, see Shpilka and Yehudayoff [2010].) As opposed to a Euclidean space such as $\mathbb{R}$, in finite fields, there is no distance measure and thus, there is no notion of "limit" and "infinitely small numbers." Hence, instead of defining partial derivatives via limits as done for Euclidean spaces, over finite fields, derivatives for polynomials and rational functions are a *formal* notion; that is, the derivative of x^n is defined to be equal to nx^{n-1} and this definition is extended to polynomials by additivity and to rational functions by using the quotient rule for derivatives. Throughout the article, we will use formal derivatives only over sufficiently large fields whose characteristic is strictly larger than the degree of the underlying polynomial. It turns out that in this setting, formal derivatives inherit many of the properties of "standard derivatives."[14]

[14]For example, in this case, a polynomial is constant if and only if its derivative is the all-zero function. This rule does not apply when the degree exceeds the field's characteristic, as demonstrated by the nonzero polynomial $x^{|\mathbb{F}|}$ whose formal derivative $|\mathbb{F}|x^{|\mathbb{F}|-1}$ is the zero function.

Definition 3.4 (*Partial Derivative*). For a finite field $\mathbb{F}$ and monomial $M(x) = ax_1^{c_1} \cdots x_n^{c_n}$ with $a \in \mathbb{F}$ and for all $1 \le i \le n$, the (formal) partial derivative with respect to x_i is defined as the monomial

$$\partial_{x_i} M(x) := c_i a \cdot x_1^{c_1} \cdots x_i^{c_i - 1} \cdots x_n^{c_n}.$$

The partial derivative of a polynomial is defined as the sum of the derivatives of its monomials. The partial derivative of a rational function $\frac{v(x)}{u(x)}$ is defined by

$$\partial_{x_i}\left(\frac{v(x)}{u(x)}\right) := \frac{u(x)\partial_{x_i} v(x) - v(x)\partial_{x_i} u(x)}{(u(x))^2}.$$

Notice that the partial derivative of a polynomial (rational function, respectively) is also a polynomial (rational function, respectively). It can be verified that the derivatives of two equivalent rational functions are also equivalent. For $f(x)$, a vector of ℓ rational functions in n variables $(f_1(x), \ldots, f_\ell(x))$, we denote by $\partial_{x_i} f(x)$ the column vector $(\partial_{x_i} f_1(x), \ldots, \partial_{x_i} f_\ell(x))^T$. In line with using normal font x for variables and bold font $\boldsymbol{x}$ for a specific point, the notation $\partial_{x_i} f(\boldsymbol{x})$ refers to the partial derivative of $f(x)$ with respect to x_i evaluated at the point $\boldsymbol{x}$.

Definition 3.5 (*Formal Jacobian*). For $f(x) = (f_1(x), \ldots, f_\ell(x))$, a vector of ℓ rational functions in n variables over a finite field $\mathbb{F}$, the Jacobian matrix $\mathcal{J} f(x)$ is the $\ell \times n$ matrix of formal derivatives $[\partial_{x_1} f(x), \ldots, \partial_{x_n} f(x)]$; that is, the (i, j)-th entry of $\mathcal{J} f(x)$ is $\partial_{x_j} f_i(x)$.

By $\mathcal{J} f(\boldsymbol{x})$, we denote all partial derivatives when evaluated at the same point $\boldsymbol{x}$. For a subset $s \subset \{x_1, \ldots, x_n\}$ of input variables, we let $\mathcal{J}_s f(x)$ denote the submatrix of $\mathcal{J} f(x)$ that contains only the columns that correspond to the variables in s.

The standard product rule and chain rule also apply for formal derivative of rational functions over finite fields.

FACT 3.6 (PRODUCT RULE FOR RATIONAL FUNCTIONS). *Let $f(x)$ and $g(x)$ be two rational functions in n variables over a finite field $\mathbb{F}$. Then, for all $1 \le i \le n$, it holds that*

$$\partial_{x_i}(f(x) \cdot g(x)) \equiv g(x)\partial_{x_i} f(x) + f(x)\partial_{x_i} g(x).$$

FACT 3.7 (CHAIN RULE). *Let $g(x) = (g_1(x), \ldots, g_\ell(x))$ be a vector of rational functions in n variables over a finite field $\mathbb{F}$, and let $f(y) = (f_1(y), \ldots, f_m(y))$ be a vector of rational functions in ℓ variables over $\mathbb{F}$. Then,*

$$\mathcal{J}(f \circ g)(x) \equiv \mathcal{J} f(g(x)) \cdot \mathcal{J} g(x).$$

3.3. Efficient Field Families

Throughout the article, we consider finite fields whose elements have an efficient representation and whose field operations are efficiently computable. We begin by defining what it means for a Boolean circuit to implement a field.

Definition 3.8 (*Circuit Implementation of a Field*). Let F be a Boolean circuit that takes as an input an operation $\mathsf{op} \in \{\mathsf{add}, \mathsf{subtract}, \mathsf{multiply}, \mathsf{divide}, \mathsf{constant}, \mathsf{zerocheck}, \mathsf{sample}, \mathsf{bitsample}\}$ (using an appropriate encoding) and up to two binary strings of length ℓ. F is said to be a *valid implementation of the finite field* $\mathbb{F}$ if there is an injective mapping $\mathsf{label} : \mathbb{F} \to \{0, 1\}^\ell$ such that the following holds:

—For every operation $\mathsf{op} \in \{\mathsf{add}, \mathsf{subtract}, \mathsf{multiply}, \mathsf{divide}\}$ and any $x, y \in \mathbb{F}$, it holds that $F(\mathsf{op}, \mathsf{label}(x), \mathsf{label}(y)) = \mathsf{label}(x *_{\mathbb{F}} y)$, where $*_{\mathbb{F}}$ is the corresponding field operation.
—$F(\mathsf{constant}, 0^\ell) = \mathsf{label}(0)$, and $F(\mathsf{constant}, 1^\ell) = \mathsf{label}(1)$.
—If $a = \mathsf{label}(0)$, then $F(\mathsf{zerocheck}, a) = \mathsf{label}(1)$.

—If $a = \mathsf{label}(x)$ for $x \in \mathbb{F}$, $x \neq 0$, then $F(\mathsf{zerocheck}, a) = \mathsf{label}(0)$.
—$F(\mathsf{sample})$ implements the uniform distribution over $\{\mathsf{label}(x) : x \in \mathbb{F}\}$.
—$F(\mathsf{bitsample})$ implements the uniform distribution over $\{\mathsf{label}(x) : x \in \{0, 1\}\}$.

Definition 3.9 (*Efficient Field Family*). A polynomial-size Boolean circuit family $F = \{F_n\}$ *implements* a family of fields $\{\mathbb{F}_n\}$ if each circuit F_n implements $\mathbb{F}_n$. In the uniform setting, we say that a PPT F *implements* a field if $F(1^n)$ outputs, with all but negligible probability, a circuit F_n that implements some field (as per Definition 3.8). That is, $F(1^n)$ defines a *probability distribution* over finite fields.

We often speak of the family of (distributions over) fields $\{\mathbb{F}_n\}$ and its efficient implementation F interchangeably. Moreover, we omit the subscript n when it is clear from context. As an example of an efficient field family, for each sequence of n-bit primes $\{p_n\}_{n\in\mathbb{N}}$, we can implement the field family $\{\mathbb{F}_n = \mathrm{GF}(p_n)\}$ by a (nonuniform) Boolean circuit family $\{F_n\}$. The uniform definition captures, for example, a PPT adversary that chooses a random n-bit prime p and outputs a circuit that implements the field $\mathrm{GF}(p)$. These variants correspond to the standard distinction between uniform and nonuniform adversaries. We remark that all our results are not sensitive to uniformity issues. Specifically, our negative results hold even if the adversary is uniform, while our positive results hold both in the uniform and nonuniform adversarial model (depending on the underlying assumptions).

3.4. Arithmetic Circuits

An arithmetic circuit is a circuit whose wires carry field elements and whose gates perform arithmetic operations. The circuit is black box with respect to the field representation; that is, it does not have access to the bit representation of the elements carried by its wires. In addition to the standard arithmetic operations, we allow "checking for zero." Although not an arithmetic operation, we find it hard to design meaningful schemes when even equality cannot be recognized and so we allow for this operation in our model.

Randomized arithmetic circuits also have two special, randomized gates: The sample gate draws elements from the underlying field $\mathbb{F}$ uniformly at random, and the bitsample gate draws a uniformly random element from the set $\{0, 1\}$ that contains the zero and one elements of the field.

Definition 3.10 (*Arithmetic Circuit*). An arithmetic circuit is a directed acyclic graph. Each input gate (source) is labeled by an input variable x_i, a constant 1 or 0, or a randomized gate sample or bitsample. Internal gates are labeled by

$$\{\mathsf{add}, \mathsf{subtract}, \mathsf{multiply}, \mathsf{divide}, \mathsf{zerocheck}\}.$$

For an arithmetic circuit $C(x)$ with n input variables and m output variables, any field $\mathbb{F}$ induces in the natural way a (randomized) mapping $C^{\mathbb{F}}(x) : \mathbb{F}^n \to \mathbb{F}^m$. Furthermore, a field implementation F naturally induces a Boolean circuit $C^F(x)$, which implements the mapping $C^{\mathbb{F}}(x)$ with the representation F. We use $C^F(x)$, $C^{\mathbb{F}}(x)$ and $C(x)$ interchangeably, when F and $\mathbb{F}$ are clear from context.

Throughout this work, when discussing a family of circuits $C = \{C_n\}$, we assume by default polynomial-time uniformity. That is, there exist a PPT Turing machine that on input 1^n outputs a description of C_n.

We will occasionally consider restricted forms of arithmetic circuits that use only a subset of the types of gates. Most notably, we will consider *deterministic* arithmetic circuits (which do not use sample, bitsample gates), deterministic arithmetic circuits without zerocheck gates (which compute rational functions), and deterministic arithmetic circuits without zerocheck, divide gates (which compute polynomials). We refer to the latter (most restricted) type of circuits as *strictly arithmetic circuits*. The following fact states that one can efficiently compute derivatives for deterministic arithmetic circuits without zerocheck gates.

FACT 3.11 (EFFICIENT DIFFERENTIATION [BAUR AND STRASSEN 1983]). *Let $C(x)$ be a deterministic arithmetic circuit of size s in n variables without* zerocheck *gates, and let $f(x)$ be the corresponding rational function. Then, for all $1 \leq i \leq n$, we can construct a circuit $C'(x)$ that evaluates the partial derivative $\partial_{x_i} f(x)$ in time $O(s)$, and constructing $C'(x)$ from $C(x)$ is a polynomial-time operation.*

The derivative of a circuit with zerocheck gates is not well defined. (In fact, such a circuit may not compute a rational function.) Still, it turns out that, over large fields, such circuits can be "approximated" well by arithmetic circuits *without* zerocheck gates. Specifically, the following fact will be useful for our lower bounds.

PROPOSITION 3.12 (REMOVING zerocheck GATES). *Let $C : \mathbb{F}^n \to \mathbb{F}^m$ be a deterministic arithmetic circuit with* zerocheck *gates of size s that never divides by zero, and let $\mathbb{F} = \mathrm{GF}(p)$ be a prime-order field of cardinality larger than $(s+1)2^{s+1}$. Then, there exists a deterministic arithmetic circuit $D : \mathbb{F}^n \to \mathbb{F}^m$ without* zerocheck *gates of size s and a strictly arithmetic circuit $G : \mathbb{F}^n \to \mathbb{F}$ of size at most s^2 that computes a polynomial of degree at most $s2^{s+1}$ that satisfy the following properties:*

(*1*) *For every $\boldsymbol{x} \in \mathbb{F}^n$ that is not a root of G, it holds that $C(\boldsymbol{x}) = D(\boldsymbol{x})$.*
(*2*) *D is obtained from C be replacing the ith* zerocheck *gate with some constant gate $b_i \in \{0, 1\}$. Furthermore, the ith* zerocheck *gate of D evaluates to b_i on all the inputs $\boldsymbol{x} \in \mathbb{F}^n$ that are not roots of G.*
(*3*) *If C computes a polynomial f of degree at most 2^s, then D computes a rational function that is formally equivalent to f.*
(*4*) *There exists a probabilistic algorithm that given (C, p) outputs with probability $1-\beta$ the circuits D and G in time* $\mathrm{poly}(s, \log p, \log(1/\beta))$.

PROOF. The circuit D is defined by repeatedly applying the following subroutine: choose the first zerocheck gate (according to some topological order) and consider the function g computed by its incoming wire. If g is the zero function, replace the gate by the constant 1 ("the zero-test passes"); otherwise, replace the gate with the constant 0 ("the zero-test fails").

Observe that the function g_i considered in the ith iteration, is computed by a zerocheck-free circuit of size at most s, and therefore is a rational function of the form u_i/v_i, where the degrees of the numerator u_i and the denominator v_i are bounded by 2^s. Furthermore, observe that none of the v_i is identically zero, since otherwise, the original circuit C tries to divide by zero (when it is applied to some inputs). Using standard techniques (cf. Shpilka and Yehudayoff [2010, Proof of Thm. 2.11]), we can extract a strictly arithmetic circuit of size s that computes u_i (v_i, respectively).

Call a point $\boldsymbol{x} \in \mathbb{F}^n$ *bad* if it is a root of some *nonzero* u_i or some (nonzero) v_i, and call it *good* otherwise. Observe that D and C agree on all the good points. Moreover, the sequence of values that a good point $\boldsymbol{x}$ induces on the zerocheck gates of the original circuit C corresponds exactly to the constants used by D to replace these gates. Letting G be the product of all the u_i and v_i, we derive the first two items. Furthermore, the fourth item follows by checking if each of the u_is is identically zero via the standard Polynomial Identity Testing algorithm (e.g., by evaluating u_i on $\mathrm{poly}(\log(s/\beta))$ random points and accepting if and only if all the outcomes are equal to zero).

It is left to prove the third item. Since the u_is and v_is are low-degree polynomials, we have, by the Schwartz-Zippel Lemma and by a union bound, that all but a $s2^{s+1}/|\mathbb{F}|$ fraction of the inputs are good. By assumption, C computes a polynomial f of degree at most 2^s. Recall that D and C can be computed by a circuit of size s, and therefore the degrees of the numerator and denominator of $D = p/q$ are also upper-bounded by 2^s. It is not hard to show that such low-degree functions D and f that agree on so many (good) points must be formally equivalent. Indeed, if this is not the case, then the polynomial $f(x)q(x) - p(x)$ is a nonzero polynomial of degree 2^{s+1} with a fraction of $1 - (s2^{s+1}/|\mathbb{F}|)$ roots. Since our field is large, $|\mathbb{F}| > (s + 1)2^{s+1}$, this contradicts the Schwartz-Zippel Lemma. □

For some of our attacks, we will need the following variant of the zerocheck removal procedure.

Definition 3.13 (*The Algorithm $\mathcal{T}$*). The algorithm $\mathcal{T}$ takes as input an arithmetic circuit $C : \mathbb{F}^n \to \mathbb{F}^m$ defined over a prime field $\mathbb{F}$ and a point $\boldsymbol{x} \in \mathbb{F}^n$. The algorithm $\mathcal{T}$ evaluates the circuit C on the point $\boldsymbol{x}$ and replaces each zerocheck gate g with the constant 1 or the constant 0 depending on the value that $\boldsymbol{x}$ induces on g. At the end, it outputs the resulting zerocheck-free circuit $C'(x)$.

Imagine that the circuit $C = B \circ A$ consists of a composition of two arithmetic circuits, an inner part A and an outer part B. Further, assume that $\mathcal{T}$ is applied separately to A and B, that is, $A' = \mathcal{T}(A, \boldsymbol{x})$ and $B' = \mathcal{T}(B, A(\boldsymbol{x}))$. The following key lemma shows that for a random $\boldsymbol{x}$ and sufficiently large $\mathbb{F}$, the composed circuit $B' \circ A'$ computes f and, more importantly for our attacks, the chain rule applies to the corresponding derivatives.

LEMMA 3.14. *Let $f(x)$ be a strictly arithmetic circuit and let $A(x)$ and $B(y)$ be two deterministic arithmetic circuits such that $B(A(\boldsymbol{x}))$ agrees with the function $f(\boldsymbol{x})$ on every input $\boldsymbol{x} \in \mathbb{F}^n$, where $\mathbb{F}$ is a field of prime order $p > 2^{2s+1}$ and $s =$ max size$\{f, B \circ A\}$. Then, for every $1 \le i \le n$,*

$$\Pr_{\boldsymbol{x} \xleftarrow{R} \mathbb{F}^n} \left[\partial_{x_i} f(\boldsymbol{x}) = \mathcal{J} B'(A(\boldsymbol{x})) \cdot \partial_{x_i} A'(\boldsymbol{x})\right] \ge 1 - \frac{2s2^s}{|\mathbb{F}|}, \tag{5}$$

where $A' := \mathcal{T}(A, \boldsymbol{x})$ and $B' := \mathcal{T}(B, A(\boldsymbol{x}))$.

PROOF. Let $C(x) = B(A(x))$ be the circuit obtained by composing B on A. By Proposition 3.12, it holds that

$$\Pr_{\boldsymbol{x} \xleftarrow{R} \mathbb{F}^n} [f \equiv \mathcal{T}(C, \boldsymbol{x})] \ge 1 - \frac{2s2^s}{|\mathbb{F}|}. \tag{6}$$

Fix some $\boldsymbol{x}$ for which the function $C' = \mathcal{T}(C, \boldsymbol{x})$ is formally equivalent to f, and let $A' = \mathcal{T}(A, \boldsymbol{x})$ and $B'(B, A(\boldsymbol{x}))$. Observe that, by the definition of $\mathcal{T}$, the circuit C' can be written as $B' \circ A'$. Now, taking derivatives in Equation (6) and applying the chain rule yields that for all $1 \le i \le n$,

$$\partial_{x_i} f(x) \equiv \mathcal{J} B'(A'(x)) \cdot \partial_{x_i} A'(x). \tag{7}$$

Fix some $1 \le i \le n$. Denote by $g(x)$ the polynomial that is computed by the LHS, and by $h(x)$ the rational function computed in the RHS. Our goal is to show that $g(\boldsymbol{x}) = h(\boldsymbol{x})$ for our fixed $\boldsymbol{x}$. This boils down to showing that $h(\boldsymbol{x})$ is well defined; that is, the denominator of h does not vanish in the point $\boldsymbol{x}$. To prove this, note that (1) $B' \circ A'$ is well defined on $\boldsymbol{x}$ (since $\mathcal{T}$ guarantees that $B'(A'(\boldsymbol{x})) = f(\boldsymbol{x})$); (2) h is the derivative of $B' \circ A'$; and (3) if a rational function H is well defined on a point $\boldsymbol{x}$, then its derivative h is also well defined on $\boldsymbol{x}$. □

4. CRYPTOGRAPHIC PRIMITIVES IN THE ARITHMETIC SETTING

We define the arithmetic versions of the main cryptographic primitives studied in the work. The reader may want to skip this section for now and refer to it later when appropriate. In the following, we will let n denote the security parameter.

4.1. Pseudorandom Generators

We begin with a definition of an *arithmetic pseudorandom generator* (APRG). Our definition of APRG requires the output to be pseudorandom over $\mathbb{F}^{\ell}$ but allows the input (the seed) to contain both random field elements and random bits as long their total number is n. We discuss this choice later in Section 9.

Definition 4.1 (*Arithmetic Pseudorandom Generator*). Let $\mathsf{APRG} = \{\mathsf{APRG}_n\}$ be a sequence of polynomial-sized arithmetic circuits where APRG_n outputs a vector of $\ell(n)$ field elements and uses at most n random gates (some of them may be for random field elements and some may be for sampling random bits) and no deterministic input gates. We say that APRG is an *arithmetic pseudorandom generator* if it satisfies the following two properties:

(1) Expansion: $\ell > n$. We refer to the ratio $\frac{\ell}{n}$ as the *expansion factor* and to the difference $\ell - n$ as the *additive expansion*.
(2) Pseudorandomness: The winning probability of every efficient adversary $\mathcal{A}$ in the following pseudorandomness game is at most $1/2 + \text{neg}(n)$:
    (a) Given a security parameter 1^n, the adversary $\mathcal{A}$ picks a field implementation $\mathbb{F}$ and sends it to the challenger.
    (b) The challenger samples $b \xleftarrow{R} \{0, 1\}$.
        i. If $b = 0$: the challenger sends to the adversary a sample $y \xleftarrow{R} \mathsf{APRG}_n$.
        ii. If $b = 1$: the challenger sends to the adversary a sample $u \xleftarrow{R} \mathbb{F}^{\ell}$.
    (c) The adversary outputs b' and wins if $b' = b$.

An APRG is *simple* if it does not contain division or $\mathsf{zerocheck}$ gates (but may use random bits). We will sometimes view the APRG as a mapping from the *seed* s (i.e., randomness sampled by the random gates) to the output, denoted by $\mathsf{APRG}(s)$. Note that the total length of s is n and it may consist of random field elements and random bits.

4.2. Encryption Schemes

We define the arithmetic version of an encryption scheme. Following Goldreich [2004, Chapter 5], we treat the public-key setting and the symmetric-key setting in a unified way.

Definition 4.2 (*Arithmetic Encryption Scheme*). Let $\mathsf{KGen} = \{\mathsf{KGen}_n\}$, $\mathsf{Enc} = \{\mathsf{Enc}_n\}$, and $\mathsf{Dec} = \{\mathsf{Dec}_n\}$ be a uniform sequence of poly(n)-sized arithmetic circuits. $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is an arithmetic public-key encryption scheme if:

(1) Correctness: For every field $\mathbb{F}$, and for every message $x \in \mathbb{F}$,

$$\Pr_{(\mathsf{sk},\mathsf{pk}) \xleftarrow{R} \mathsf{KGen}_n} \left[\mathsf{Dec}_n^{\mathbb{F}}\left(\mathsf{Enc}_n^{\mathbb{F}}(x, \mathsf{pk}), \mathsf{sk}\right) = x\right] \geq 1 - \text{neg}(n),$$

where the probability is taken over the choice of keys and the randomness of the circuits $\mathsf{Enc}_n^{\mathbb{F}}$, $\mathsf{Dec}_n^{\mathbb{F}}$.
(2) Computational security: For every efficient adversary $\mathcal{A}$, the winning probability in the following IND-CPA game is at most $1/2 + \text{neg}(n)$:

IND-CPA-Game(1^n):
—The adversary receives 1^n, chooses a field $\mathbb{F}$, and sends $\mathbb{F}$ to challenger.
—The challenger samples $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$ and passes pk to the adversary.
—(Chosen plaintext queries:) Repeatedly do as long as adversary requests:
  —The adversary sends some $x \in \mathbb{F}$.
  —The challenger responds with $\mathsf{Enc}_n^{\mathbb{F}}(x, \mathsf{pk})$.
—The adversary sends some pair $x_0, x_1 \in \mathbb{F}$.
—The challenger samples $b \xleftarrow{R} \{0, 1\}$ and responds with $c \xleftarrow{R} \mathsf{Enc}_n^{\mathbb{F}}(x_b, \mathsf{pk})$.
—The adversary outputs b' and wins if $b' = b$.

The symmetric-key setting corresponds to the case where $\mathsf{pk} = \mathsf{sk}$ and the challenger does not pass pk to the adversary. The scheme is *one-time secure* if it is secure against adversaries that make no chosen plaintext queries. An arithmetic symmetric encryption scheme with one-time security is called *arithmetic/binary encryption* if all the elements of the secret key sk are taken from the subset $\{0, 1\} \subset \mathbb{F}$. That is, the key is essentially a string of bits given as a sequence of 0-1 field elements.

Remark 4.3 (*Encrypting Long Messages*). The previous definition assumes that the message contains a single field element. One could naturally extend the definition to support longer vectors (either with fixed block-length ℓ or with unbounded block length). We note that, as in the binary setting, a CPA-secure construction that supports a single message can be easily extended to encrypt a sequence of field elements by encrypting each element separately each time with fresh randomness (cf. Katz and Lindell [2008, Section 3.4]).

4.3. Commitments

We consider noninteractive commitments. Such schemes are parameterized by a public key pk, which is chosen by some trusted party or given as part of a Common Reference String (CRS).

Definition 4.4 (*Statistically Binding Commitment Scheme*). Let $\mathsf{KGen} = \{\mathsf{KGen}_n\}$, $\mathsf{Com} = \{\mathsf{Com}_n\}$, and $\mathsf{Ver} = \{\mathsf{Ver}_n\}$ be a uniform sequence of polynomial-sized arithmetic circuits. We denote the output of the circuit Com_n by (c, d), where c is the commitment and d the decommitment. We say $(\mathsf{KGen}, \mathsf{Com}, \mathsf{Ver})$ is a statistically binding commitment scheme if:

(1) Correctness: For every field $\mathbb{F}$ and for every message $x \in \mathbb{F}$, it holds that

$$\Pr_{\mathsf{pk} \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}} \left[\mathsf{Ver}_n^{\mathbb{F}}(\mathsf{pk}, x, \mathsf{Com}_n^{\mathbb{F}}(x, \mathsf{pk}))\right] \geq 1 - \mathrm{neg}(n),$$

where the probability is also taken over the randomness of the circuits $\mathsf{Com}_n^{\mathbb{F}}$ and $\mathsf{Ver}_n^{\mathbb{F}}$.

(2) Statistically binding: For every field $\mathbb{F}$, with overwhelming probability over the choice of $\mathsf{pk} \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$, no commitment c can be opened in two different ways, that is,

$$\forall x, x', c, d, d', \quad \left(\mathsf{Ver}_n^{\mathbb{F}}(\mathsf{pk}, x, c, d) = 1\right) \wedge \left(\mathsf{Ver}_n^{\mathbb{F}}(\mathsf{pk}, x', c, d') = 1\right) \Rightarrow x = x'.$$

(3) Computationally hiding: No efficient adversary $\mathcal{A}$ can win the following game with probability greater than $1/2 + \mathrm{neg}(n)$:
IND-CPA Game(1^n):
—$\mathcal{A}$ receives 1^n, chooses a field $\mathbb{F}$, and sends $\mathbb{F}$ to the challenger.
—The challenger samples $\mathsf{pk} \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$ and passes pk to the adversary.

—The adversary chooses $x_0, x_1 \in \mathbb{F}$.
—The challenger samples $b \in \{0, 1\}$, computes $(c, d) \xleftarrow{R} \mathsf{Com}_n^{\mathbb{F}}(x_b, \mathsf{pk})$, and sends c.
—The adversary outputs b' and wins if $b' = b$.

Without loss of generality, we assume that the decommittment string d consists of the internal randomness of the committer (which is a sequence of field elements and zero-one values). Under this convention, we let the output of $\mathsf{Com}_n^{\mathbb{F}}$ denote only the commitment string c.

Remark 4.5 (*CRS-free Variant*). We will also consider a CRS-free variant of commitments. Syntactically, we still consider a noninteractive commitment function parameterized by a public key that is generated by a key-generation algorithm. However, the hiding property should hold for every (adversarially chosen) public key. This gives rise to a two-message commitment protocol in the standard model (no CRS) in which the receiver chooses the public key pk.

4.4. Randomized Encoding of Functions

We formalize arithmetic garbled circuits via the framework of *randomized encoding* of functions [Ishai and Kushilevitz 2000; Applebaum et al. 2004]. Roughly speaking, a deterministic function f is encoded by a randomized arithmetic function $\hat{f}$ if for every input x, the distribution $\hat{f}(x)$, induced by the internal randomness of $\hat{f}$, reveals the value of $f(x)$ and nothing else.

Definition 4.6 (*Arithmetic Randomized Encoding*). Let $f = \{f_n\}$ be a family of polynomial-sized strictly arithmetic circuits where f_n has n inputs. A family of polynomial-sized randomized arithmetic circuits $\hat{f} = \{\hat{f}_n\}$ and a family of polynomial-sized deterministic arithmetic circuits $\mathsf{Dec} = \{\mathsf{Dec}_n\}$ form a *Randomized Encoding* of f if:

(1) Perfect correctness: For every field $\mathbb{F}$, every $n \in \mathbb{N}$, and every $\boldsymbol{x} \in \mathbb{F}^n$,

$$\Pr\left[\mathsf{Dec}_n^{\mathbb{F}}\left(\hat{f}_n^{\mathbb{F}}(\boldsymbol{x})\right) = f_n^{\mathbb{F}}(\boldsymbol{x})\right] = 1,$$

where the probability is taken over the randomness of the encoding circuit $\hat{f}_n^{\mathbb{F}}$.
(2) Computational security (indistinguishability):
For any PPT $\mathcal{A}(1^n)$, the winning probability in the following game is at most $1/2 + \mathrm{neg}(n)$:
—The adversary $\mathcal{A}$ chooses an implementation of a field $\mathbb{F}$ and $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathbb{F}^n$ s.t. $f_n^{\mathbb{F}}(\boldsymbol{x}_0) = f_n^{\mathbb{F}}(\boldsymbol{x}_1)$ and sends to the challenger $\mathbb{F}, \boldsymbol{x}_0, \boldsymbol{x}_1$.
—The challenger samples $b \xleftarrow{R} \{0, 1\}$ and $c = \hat{f}_n^{\mathbb{F}}(\boldsymbol{x}_b)$ and sends c to $\mathcal{A}$.
—The adversary $\mathcal{A}$ outputs a bit b' and wins if $b' = b$.

Remark 4.7. We mention that security is typically formalized via a stronger simulation-based definition. However, since we will be proving lower bounds, a weaker security definition only makes our results stronger. Also, note that we restrict the encoded function f to be computed by a strictly arithmetic circuit (i.e., by an arithmetic circuit over addition and multiplication gates only), whereas the encoding $\hat{f}$ is allowed to be an arbitrary arithmetic circuit. Again, this only makes our lower bounds stronger.

Without any efficiency restriction on the encoder $\hat{f}$, the notion of randomized encoding becomes trivial (simply take $\hat{f} = f$). Next we present several common syntactic and efficiency requirements that are motivated by different RE applications.

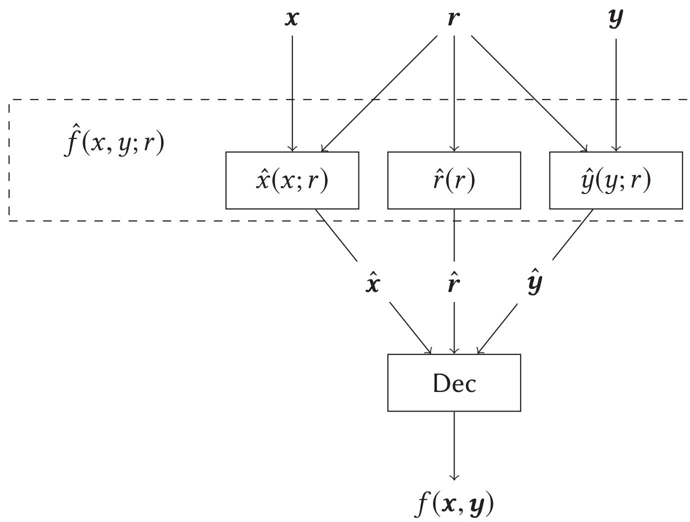


Fig. 1. 2-Decomposable randomized encoding.

Decomposable encoding. For many applications, it makes sense to distinguish between the online part, the part of the output of $\hat{f}$ that depends on the input vector x, and the offline part, the part of the output that depends only on the randomness r of the encoding. We will emphasize this distinction by splitting $\hat{f}(x;r)$ into the functions $\hat{x}(x;r)$ and $\hat{r}(r)$, where r denotes the outcome of the random gates. A randomized encoding $(\hat{f}, \mathsf{Dec})$ is *fully decomposable* if each of the outputs of the encoder $\hat{f}(x;r)$ depends on at most a single element in the input vector x. In such a case, we will write $\hat{x}(x;r)$ as $(\hat{x}_1(x_1;r), \ldots, \hat{x}_n(x_n;r))$ to emphasize which part of the encoding depends on which input. One may define a weaker form of decomposability. Suppose that the encoded function f is defined over two input vectors x and y; then an encoding $(\hat{f}, \mathsf{Dec})$ is *2-decomposable* if each output of $\hat{f}(x,y;r)$ depends either on x or on y but not on both (but may depend arbitrarily on r). In this case, we will usually denote $\hat{f}(x,y;r) = (\hat{x}(x;r), \hat{y}(y;r), \hat{r}(r))$ to clarify which part of the encoding depends on which input. This notion corresponds to the PSM model described at the introduction.[15] An illustration of 2-decomposable encoding is depicted in Figure 1.

4.5. Secure Computation

Arithmetic functionalities. A two-party arithmetic functionality $f : \mathbb{F}^* \times \mathbb{F}^* \to \mathbb{F}^* \times \mathbb{F}^*$ is described by a sequence of polynomial-size arithmetic circuits $\{f_n\}$ where each input and output gate is labeled by A (for Alice) or by B (for Bob). For simplicity, we will consider only deterministic functionalities and always assume that in f_n, both Alice and Bob have exactly n inputs. Both assumptions hold without loss of generality. (See Goldreich [2004], Proposition 7.3.4. and Section 7.2.1.1.) For a field implementation $\mathbb{F}$ and pair of inputs $x, y \in \mathbb{F}^n$ for Alice and Bob, we let $f_1^{\mathbb{F}}(x,y)$ ($f_2^{\mathbb{F}}(x,y)$, respectively) denote the outputs of Alice (Bob, respectively).

Arithmetic protocols. A two-party arithmetic protocol Π is a pair of (uniform) polynomial-size arithmetic circuits $A = \{A_n\}$ and $B = \{B_n\}$, indexed by input length n.

[15]To see this, let r be the randomness shared between Alice and Bob, and let Alice (Bob, respectively) send $\hat{x}(x;r)$ ($\hat{y}(y;r)$, respectively) to Carol. In addition, the value $\hat{r}(r)$ can be sent to Carol in an offline phase or by one of the parties. Given all these values, Carol can learn $f(x,y)$ (using the decoder) without learning anything else.

The circuits A_n and B_n interact via special "interaction gates" Send and Receive. Send gates have unbounded fan-in and no fan-out, and Receive gates have an unbounded fan-out and zero fan-in. These gates are assumed to be topologically ordered. We assume that the gates are aligned so that in A_n, odd-numbered interaction gates are all Send gates and even-numbered interaction gates are Receive gates, and vice versa for B_n. Semantically, we assume that the ith Send gate of a party (say, A) corresponds to the ith Receive gate of the other party; that is, if $x \in \mathbb{F}^*$ is the input to Send, then Receive $= x$. For every field implementation $\mathbb{F}_n$, the protocol naturally defines a joint computation that maps a pair of inputs $(x, y) \in \mathbb{F}_n^n \times \mathbb{F}_n^n$ to a pair of outputs $\mathsf{Output}^{\Pi,\mathbb{F}_n}(x, y) = (\mathsf{Output}_1, \mathsf{Output}_2)$, where Output_1 is the output of A_n and Output_2 is the output of B_n. The *view* of A during an execution of Π over $\mathbb{F}_n$ on inputs $x, y \in \mathbb{F}^n$, denoted by $\mathsf{View}_1^{\Pi,\mathbb{F}}(x, y)$, is the random variable that consists of the deterministic and random inputs of A_n and all the messages sent by B_n. The view of B is defined similarly and is denoted by $\mathsf{View}_2^{\Pi,\mathbb{F}}(x, y)$.

4.5.1. The Semihonest Model.

Definition 4.8 (*Semihonest Secure Computation (Deterministic Functionalities)*). We say that a two-party arithmetic protocol $\Pi = (\{A_n\}, \{B_n\})$ *privately computes* a (deterministic) functionality $f : \mathbb{F}^* \times \mathbb{F}^* \to \mathbb{F}^* \times \mathbb{F}^*$ if the following hold for every efficient field family $\{\mathbb{F}_n\}$:

—Correctness: For every $x, y \in \mathbb{F}_n^n$,

$$\Pr\left[\mathsf{Output}^{\Pi,\mathbb{F}_n}(x, y) \neq f^{\mathbb{F}_n}(x, y)\right] \leq \mathrm{neg}(n).$$

—Privacy: There exist probabilistic polynomial-time algorithms Sim_1 and Sim_2 such that

$$\left\{\mathsf{View}_1^{\Pi,\mathbb{F}_n}(x, y)\right\}_{x,y\in\mathbb{F}_n^n} \stackrel{c}{\approx} \left\{\mathsf{Sim}_1\big(x,\ f_1^{\mathbb{F}_n}(x, y)\big)\right\}_{x,y\in\mathbb{F}_n^n}$$

and

$$\left\{\mathsf{View}_2^{\Pi,\mathbb{F}_n}(x, y)\right\}_{x,y\in\mathbb{F}_n^n} \stackrel{c}{\approx} \left\{\mathsf{Sim}_2\big(y,\ f_2^{\mathbb{F}_n}(x, y)\big)\right\}_{x,y\in\mathbb{F}_n^n}.$$

Note that the definition allows the simulators to depend on the field family $\{\mathbb{F}_n\}$ in a non-black-box way. Nevertheless, in all our constructions, the simulators will also be defined by arithmetic circuits and will access the underlying field $\mathbb{F}$ only in a black-box way. For our lower bounds, we will consider a weaker version of *One-Sided* Semihonest Security (see Section 6 for details).

Partial functionalities, finite functionalities, and different input lengths. We sometimes consider partial functionalities (e.g., when some of the inputs are binary strings) and in this case the aforementioned requirements should be adopted by indexing the ensembles by appropriate inputs. Furthermore, we will use finite functionalities (e.g., oblivious transfer) and in this case incorporate the security parameter 1^n as part of the parties' input. We will also consider functionalities in which the input length of Alice may be different from the input length of Bob, with the implicit convention that we can always guarantee equal input length by padding.

Privacy reductions. To define secure reductions, consider the following *hybrid* model. A two-party protocol augmented with an oracle to the two-party arithmetic functionality g is a standard protocol in which the parties are allowed to invoke g, that is, a trusted party to which they can securely send inputs and receive the corresponding outputs. Formally, the circuits A_n and B_n are equipped with an equal number of special g gates, where the ith g-gate takes an input a_i from A and an input b_i from B and returns an output $g_1(a, b)$ to A and $g_2(a, b)$ to B. When A_n and B_n are instantiated with a field $\mathbb{F}$, the functionality g is instantiated with the same field $\mathbb{F}$ as well. The notion of private computation (Definition 4.8) generalizes to protocols augmented with an oracle in the natural way.

A *private reduction* from an arithmetic two-party functionality f to an arithmetic two-party functionality g is a two-party protocol that, given an oracle access to the functionality g, privately realizes the functionality f. Appropriate composition theorems (e.g. Goldreich [2004, Thms. 7.3.3, 7.4.3]) guarantee that the call to g can be replaced by any private protocol realizing g, without violating the privacy of the high-level protocol for f.

4.5.2. The Malicious Model. We define arithmetic secure two-party computation in the malicious setting using the *Real-Ideal* paradigm. Following our general convention, an honest party in this model is an arithmetic circuit, while a malicious party is allowed to be any polynomial-size binary circuit. Formally, we say that an arithmetic protocol (A, B) *securely realizes* an arithmetic (possibly randomized) two-party functionality f in the malicious setting (in short, (A, B) *securely realizes* f) if, for any efficient field family $\mathbb{F} = \{\mathbb{F}_n\}$, the binary protocol $(A, B)^{\mathbb{F}}$ securely realizes the *binary* two-party functionality $f^{\mathbb{F}}$. For completeness, we briefly review the standard definitions of maliciously secure two-computation in the binary model. (See Goldreich [2004, Chapter 7] for more detailed and concrete definitions.)

Let $f(x, y)$ be a two-party functionality, that is, a (possibly randomized) mapping from a pair of inputs of equal length into a pair of outputs. Let π be a two-party protocol. We formulate the requirement that π securely computes f by comparing the following "real process" and "ideal process."

The real process. An adversary A^* attacking Alice in the *real process* is a family of probabilistic polynomial-size circuits who observe all the internal data of Alice and, in addition, has full control over the messages sent by Alice. At the end of the interaction, the adversary may output an arbitrary function of its view, which consists of the inputs, the random coin tosses, and the incoming messages of the corrupted parties. Given a pair of inputs $(x, y) \in (\{0, 1\}^n)^2$ and an auxiliary input for the adversary $\mathsf{aux} \in \{0, 1\}^*$, the *output of the real process* is defined as the random variable containing the *concatenation* of the adversary's output with the outputs and identities of the uncorrupted party. We denote this output by $\mathsf{Real}_{\pi, A^*(\mathsf{aux})}(x, y)$. The case of an adversary B^* that attacks Bob is defined similarly and in this case we let $\mathsf{Real}_{\pi, B^*(\mathsf{aux})}(x, y)$ denote the output of the real process.

The ideal process. In the ideal process, an incorruptible trusted party is employed for computing the given functionality. That is, the "protocol" in the ideal process instructs each party to send its input to the trusted party, who computes the functionality f and sends to each party its output. The interaction of an adversary Sim_A, which attacks Alice (Sim_B, which attacks B, respectively), with the ideal process and the output of the ideal process is defined analogously to the previous definitions for the real process, except that we allow the adversary to send a special "abort" message (possibly after receiving his output) and, in this case, the trusted party sends an abort message to the honest party, who halts with a special abort symbol. The adversary attacking the ideal process will also be referred to as a *simulator*. We denote the output of the ideal process on the inputs $(x, y) \in (\{0, 1\}^n)^2$ and auxiliary input aux by $\mathsf{Ideal}_{f, \mathsf{Sim}_A(\mathsf{aux})}(x, y)$ ($\mathsf{Ideal}_{f, \mathsf{Sim}_B(\mathsf{aux})}(x, y)$, respectively).

Definition 4.9 (*Maliciously Secure Two-party Computation*). The protocol π is said to securely realize the given functionality f if for any probabilistic polynomial-size circuit A^* attacking Alice (B^* attacking Bob, respectively) in the real process there exists a probabilistic polynomial-size circuit simulator Sim_A attacking Alice (Sim_B attacking Bob, respectively) in the ideal process, such that

$$\left\{\mathsf{Real}_{\pi,A^*(\mathsf{aux})}(x,y)\right\}_{x,y\in\{0,1\}^n,\mathsf{aux}\in\{0,1\}^{\mathrm{poly}(n)}} \stackrel{c}{\equiv} \left\{\mathsf{Ideal}_{f,\mathsf{Sim}_A(\mathsf{aux})}(x,y)\right\}_{x,y\in\{0,1\}^n,\mathsf{aux}\in\{0,1\}^{\mathrm{poly}(n)}},$$

and similarly for Bob:

$$\left\{\mathsf{Real}_{\pi,B^*(\mathsf{aux})}(x,y)\right\}_{x,y\in\{0,1\}^n,\mathsf{aux}\in\{0,1\}^{\mathrm{poly}(n)}} \stackrel{c}{\equiv} \left\{\mathsf{Ideal}_{f,\mathsf{Sim}_B(\mathsf{aux})}(x,y)\right\}_{x,y\in\{0,1\}^n,\mathsf{aux}\in\{0,1\}^{\mathrm{poly}(n)}}.$$

An arithmetic protocol (A, B) *securely realizes* an arithmetic (possibly randomized) two-party functionality f if for any efficient field family $\mathbb{F} = \{\mathbb{F}_n\}$ the binary protocol $(A, B)^{\mathbb{F}}$ securely realizes the binary two-party functionality $f^{\mathbb{F}}$.

Secure reductions (in the malicious model) are defined analogously to private reductions (in the semihonest model).

LOWER BOUNDS

In the following sections, we will show that some cryptographic primitives cannot be realized in the arithmetic setting. Section 5 is dedicated to Garbed Circuits (formalized under the framework of Randomized Encodings), Section 6 discusses communication lower bounds for Secure Computation protocols, and Section 7 is devoted to Homomorphic Encryption.

5. GARBLED CIRCUITS AND RANDOMIZED ENCODING

Recall that a deterministic function $f(x, y)$ is encoded by a two-decomposable randomized arithmetic function $\hat{f}(x, y; r) = (\hat{x}(x; r), \hat{y}(y; r), \hat{r}(r))$ if for every input (x, y) the distribution $\hat{f}(x, y; r)$, induced by a random choice of the internal randomness r of $\hat{f}$, reveals the value of $f(x, y)$ and nothing else. (See Section 4.4 for relevant definitions.)

We show that an arithmetic 2-decomposable encoding must have large communication. In particular, we prove (in Section 5.1) that the encoding $|\hat{x}|$ of x needs to be as long as the output $|f(x, y)|$.

THEOREM 5.1. *There exists an arithmetic function $f_n(x, y)$ that maps $x \in \mathbb{F}$ and $y \in \mathbb{F}^{2n}$ to an output in $\mathbb{F}^n$ for which in any 2-decomposable arithmetic randomized encoding $\hat{f}_n(x, y; r) = (\hat{x}(x; r), \hat{y}(y; r), \hat{r}(r))$ of f, the function $\hat{x}$ consists of at least n field elements.*

In contrast, in the binary setting, Yao's garbled circuit construction [Yao 1986] provides, for any efficiently computable function $f(x, y)$, a 2-decomposable encoding $\hat{f}_n(x, y; r) = (\hat{x}(x; r), \hat{y}(y; r), \hat{r}(r))$ with $|\hat{x}| = |x| \cdot \kappa$, where κ is the security parameter that can be taken to n^{ε} for an arbitrary small constant $\varepsilon > 0$.

For fully decomposable encoding, we prove (in Section 5.2) that the total online complexity $|\hat{x}|$ needs to be roughly as long as the product of the input length and the output length.

THEOREM 5.2. *Assuming the existence of a binary* PRG*, there exists an arithmetic function $f : \mathbb{F}^{3n} \to \mathbb{F}^n$ for which any fully decomposable arithmetic randomized encoding*

$$\hat{f}(x; r) = (\hat{x}_1(x_1; r), \ldots, \hat{x}_{3n}(x_{3n}; r), \hat{r}(r))$$

has online complexity $\sum_i |\hat{x}_i|$ of at least n^2.

Again, in the binary setting, Yao's garbled circuit construction provides, for any efficiently computable function $f(x)$, a fully decomposable encoding $\hat{f}(x; r) =$

$(\hat{x}_1(x_1; r), \ldots, \hat{x}_n(x_{3n}; r), \hat{r}(r))$ with online complexity $\sum_i |\hat{x}_i| = |x| \cdot n^{\varepsilon}$ for arbitrary small constant $\varepsilon > 0$.

5.1. Proof of Theorem 5.1

Overview. We follow the outline sketched in the introduction. Roughly speaking, we will exploit the correctness property $f(x) = \mathsf{Dec}(\hat{f}(x; r))$ to argue that, over sufficiently large field $\mathbb{F}$, for every fixing of the randomness $\boldsymbol{r}$ the circuits f and $\mathsf{Dec} \circ \hat{f}(\cdot; \boldsymbol{r})$ compute the same rational functions. Therefore, by the chain rule, we have that for any input variable x_i,

$$\partial_{x_i} f(x) \equiv \mathcal{J}\mathsf{Dec}(\hat{f}(x; \boldsymbol{r})) \cdot \partial_{x_i} \hat{f}(x; \boldsymbol{r}).$$

Hence, given Dec and an encoding $\hat{f}(\boldsymbol{x}; \boldsymbol{r})$ of some point $\boldsymbol{x}$, we can deduce that $\partial_{x_i} f(\boldsymbol{x})$ is spanned by the columns of $\mathcal{J}\mathsf{Dec}(\hat{f}(\boldsymbol{x}; \boldsymbol{r}))$. We will show that if the communication is small, this information violates privacy (for a properly chosen function f). This argument assumes that $\hat{f}$ and Dec compute rational functions and so their derivatives are well defined. This is not the case when the encoder $\hat{f}$ or decoder Dec uses $\mathsf{zerocheck}$ gates. To cope with this, we apply the robust $\mathsf{zerocheck}$-removal algorithm $\mathcal{T}$ from Definition 3.13.

We proceed with a formal proof of Theorem 5.1.

PROOF OF THEOREM 5.1. Consider the function $f : \mathbb{F} \times (\mathbb{F}^n \times \mathbb{F}^n) \to \mathbb{F}^n$ that maps $(x, (y, z))$ to $xy + z$, where $x \in \mathbb{F}$ is a scalar and y and z are vectors of length n. Assume toward contradiction that $\hat{f}(x, (y, z); r) = (\hat{x}, \hat{y}, \hat{r})$ and Dec forms a 2-decomposable arithmetic RE with $|\hat{x}| = m < n$. (Note that $\hat{y}$ depends on (y, z) and r.) We will show how to break the security of the RE via the following adversary. (From now on, we will often omit the underlying field $\mathbb{F}$ and the security parameter n when clear from the context, that is, $f(x) := f_n^{\mathbb{F}}(x)$.)

Adversary $\mathcal{A}(1^n)$:

- Choose a prime $p > 2^{2s+1}$ for $s =$ max size$\{f_n, \mathsf{Dec}_n \circ \hat{f}_n\}$.
- Let $\mathbb{F} = \mathrm{GF}(p)$ and send some standard implementation of $\mathbb{F}$ to the challenger.
- Sample $\boldsymbol{x}_0 \stackrel{R}{\leftarrow} \mathbb{F}$ together with $\boldsymbol{y}_0, \boldsymbol{z}_0 \stackrel{R}{\leftarrow} \mathbb{F}^n$.
- Sample $\boldsymbol{x}_1 \stackrel{R}{\leftarrow} \mathbb{F}$ together with $\boldsymbol{y}_1 \stackrel{R}{\leftarrow} \mathbb{F}^n$ and compute $\boldsymbol{z}_1 = \boldsymbol{x}_0 \boldsymbol{y}_0 + \boldsymbol{z}_0 - \boldsymbol{x}_1 \boldsymbol{y}_1$.
- Send $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{z}_0)$ and $(\boldsymbol{x}_1, \boldsymbol{y}_1, \boldsymbol{z}_1)$ to the challenger.
- Given an encoding $\boldsymbol{c} = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{r}})$, define the circuit $B(\hat{x}) = \mathsf{Dec}_n(\hat{x}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{r}})$.
- Compute the $\mathsf{zerocheck}$-free arithmetic circuit $B' = \mathcal{T}(B, \hat{\boldsymbol{x}})$.
- Output 1 if $\boldsymbol{y}_1$ is spanned by the columns of $\mathcal{J}B'(\hat{\boldsymbol{x}})$; otherwise, output 0.

Observe that this is a valid adversary as $f(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{z}_0) = f(\boldsymbol{x}_1, \boldsymbol{y}_1, \boldsymbol{z}_1)$. The following claims show that $\mathcal{A}$ distinguishes between the two cases with advantage of at least $\frac{3}{4} - \mathrm{neg}(n)$.

CLAIM 5.3. *If the challenge bit $b = 1$, that is, the challenger sends a sample $\boldsymbol{c} = \hat{f}(\boldsymbol{x}_1, \boldsymbol{y}_1, \boldsymbol{z}_1; \boldsymbol{r})$ for randomly chosen $\boldsymbol{r}$, then the adversary accepts with probability* $1 - \mathrm{neg}(n)$.

PROOF. Fix the randomness of the challenger to some value $\boldsymbol{r}$ and fix the values $\boldsymbol{y}_1, \boldsymbol{z}_1$. Let $f_{\boldsymbol{y}_1, \boldsymbol{z}_1}(x) = f_n(x, \boldsymbol{y}_1, \boldsymbol{z}_1)$, let $A(x)$ denote the circuit $\hat{f}_n(x, \boldsymbol{y}_1, \boldsymbol{z}_1; \boldsymbol{r})$, and let $B(\hat{x}) = \mathsf{Dec}_n(\hat{x}, \hat{\boldsymbol{y}}_1, \hat{\boldsymbol{r}}_1)$, where $\hat{\boldsymbol{y}}_1$ and $\hat{\boldsymbol{r}}$ are the outcome of $\hat{f}$ when applied to $\boldsymbol{y}_1, \boldsymbol{z}_1$, and $\boldsymbol{r}$. (Due to the decomposability, these values are independent of $\hat{x}$.) Syntactically,

$$f_{\boldsymbol{y}_1, \boldsymbol{z}_1} : \mathbb{F} \to \mathbb{F}^n, \qquad A : \mathbb{F} \to \mathbb{F}^m, \qquad B : \mathbb{F}^m \to \mathbb{F}^n.$$

By the perfect correctness and the decomposability property, it holds that $B(A(x)) = f_{\boldsymbol{y}_1,\boldsymbol{z}_1}(x)$ for every x. Hence, by Lemma 3.14, we have

$$\Pr_{\boldsymbol{x}_1 \xleftarrow{R} \mathbb{F}} \left[\partial_x f_{\boldsymbol{y}_1,\boldsymbol{z}_1}(\boldsymbol{x}_1) = \mathcal{J}B'(A(\boldsymbol{x}_1)) \cdot \partial_x A'(\boldsymbol{x}_1)\right] \geq 1 - \mathrm{neg}(n),$$

where $A' = \mathcal{T}(A, \boldsymbol{x}_1)$ and $B' = \mathcal{T}(B, \hat{\boldsymbol{x}}_1)$. Since $\boldsymbol{y}_1 = \partial_x f_{\boldsymbol{y}_1,\boldsymbol{z}_1}(\boldsymbol{x}_1)$ and $\boldsymbol{c} = A(\boldsymbol{x}_1)$, we conclude that

$$\Pr_{\boldsymbol{x}_1 \xleftarrow{R} \mathbb{F}} \left[\boldsymbol{y}_1 \in \mathsf{colSpan}(\mathcal{J}B'(\boldsymbol{c}))\right] \geq 1 - \mathrm{neg}(n),$$

and the claim follows. $\square$

We move on to the case where $b = 0$.

CLAIM 5.4. *If the challenge bit $b = 0$, that is, the challenger sends a sample $\boldsymbol{c} = \hat{f}(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{z}_0; \boldsymbol{r})$ for randomly chosen $\boldsymbol{r}$, then the adversary accepts with probability at most* $1/4$.

PROOF. Fix $\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{z}_0$ as well as the randomness of the challenger and note that $\boldsymbol{y}_1$ is still uniformly distributed. Let M denote the matrix $\mathcal{J}B'(\hat{f}_r(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{z}_0))$. Since the input $\hat{x}$ of B' is of length $m < n$, the matrix M is of dimensions $n \times m$, and its columns span at most $|\mathbb{F}|^m$ vectors in $\mathbb{F}^n$. It follows that

$$\Pr_{\boldsymbol{y}_1 \xleftarrow{R} \mathbb{F}^n} [\boldsymbol{y}_1 \text{ is spanned by the columns of } M] \leq |\mathbb{F}|^{m-n},$$

and so $\mathcal{A}$ outputs 1 with probability at most $|\mathbb{F}|^{m-n}$, which is upper bounded by $1/4$ for $m < n$. $\square$

Overall, $\mathcal{A}$ wins the game with probability greater than $1 - \mathrm{neg}(n) - 1/4 > 3/4 - \mathrm{neg}(n)$, and the theorem follows.

5.2. Proof of Theorem 5.2

Let $\mathsf{PRG} = \{\mathsf{PRG}_n\}$ with $\mathsf{PRG}_n : \{0, 1\}^n \to \{0, 1\}^{n^2}$ be a pseudorandom generator over the binary field. We can transform the Boolean circuit that computes it into an arithmetic circuit by substituting $\mathrm{AND}(w_1, w_2)$ gates with $w_1 \times w_2$ and $\mathrm{NOT}(w)$ gates by $1 - w$. By abuse of notation, we let $\mathsf{PRG}_n : \mathbb{F}^n \to \mathbb{F}^{n^2}$ also denote the transformed circuit. Observe that when the input to the circuit consists of $0-1$ values, the functionality is preserved. Specifically, for a random $\boldsymbol{x} \in \{0, 1\}^n$, the vector $\mathsf{PRG}(\boldsymbol{x})$ consists of a pseudorandom sequence of $0 - 1$ elements.

To prove Theorem 5.2, we will consider the function $f : \mathbb{F}^n \times \mathbb{F}^n \times \mathbb{F}^n \to \mathbb{F}^n$ that maps (x, y, z) to $Yx + z$, where Y is the $n \times n$ matrix that consists of the elements generated by $\mathsf{PRG}(y)$. Assume, toward a contradiction, that f is encoded by a fully decomposable arithmetic randomized encoding

$$\hat{f}(x, y, z; r) = (\hat{x}, \hat{y}, \hat{z}, \hat{r}),$$

with decoder Dec, where $|\hat{x}|$ is shorter than n^2. Since $\hat{f}$ is fully decomposable, we can write $\hat{x} = (\hat{x}_1, \ldots, \hat{x}_n)$ and conclude, by an averaging argument, that at least some $\hat{x}_i$ is shorter than n. We will exploit this property to break the original binary PRG. Specifically, we define the following adversary $\mathcal{A}$.

Adversary $\mathcal{A}(1^n, Y \in \{0, 1\}^{n^2})$:

—Choose a prime $p > 2^{2s+1}$ for $s = \max \mathrm{size}\{f_n, \mathsf{Dec}_n \circ \hat{f}_n\}$ and let $\mathbb{F} = \mathrm{GF}(p)$.
—Parse the challenge $Y \in \{0, 1\}^{n^2}$ into columns $(Y_1, \ldots, Y_n)$.

—Sample a point $\boldsymbol{x}, \boldsymbol{z} \xleftarrow{R} \mathbb{F}^n$ at random and fix $\boldsymbol{y}$ to some default value, for example, 0^n. Sample randomness $\boldsymbol{r}$ for the encoding $\hat{f}_n$ and let $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}, \hat{\boldsymbol{r}}) = \hat{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}; \boldsymbol{r})$.
—Compute $\mathsf{Dec}' := \mathcal{T}(\mathsf{Dec}, (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}, \hat{\boldsymbol{r}}))$.
—Output 1 if $Y_i \in \mathsf{colSpan}(\mathcal{J}_{\hat{x}_i}\mathsf{Dec}'((\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}, \hat{\boldsymbol{r}})))$ for all $i = 1, \ldots, n$; otherwise, output 0.

Analysis of the adversary. We will show that the adversary $\mathcal{A}$ accepts, with probability $1 - \mathrm{neg}(n)$, a pseudorandom string $Y = \mathsf{PRG}(\boldsymbol{y})$, where $\boldsymbol{y} \xleftarrow{R} \{0, 1\}^n$. We will prove this in two steps. First, we will show that this is the case if the adversary sets $\boldsymbol{y}$ to be the seed of Y (Claim 5.5), and then we show (by relying on the security of the encoding) that this is the case even when $\boldsymbol{y}$ is set to an arbitrary value (Claim 5.6). We will later show (Claim 5.7) that a random $Y \in \{0, 1\}^{n^2}$ is accepted with probability of at most $\frac{1}{2}$ and conclude that $\mathcal{A}$ breaks the PRG.

CLAIM 5.5. *For any fixed $\boldsymbol{r}, \boldsymbol{y}, \boldsymbol{z}$, it holds, with overwhelming probability over $\boldsymbol{x}$, that for every i:*

$$Y_i \in \mathsf{colSpan}\left(\mathcal{J}_{\hat{x}_i}\mathsf{Dec}'(\hat{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}; \boldsymbol{r}))\right),$$

where $Y = \mathsf{PRG}(\boldsymbol{y})$, $\mathsf{Dec}' := \mathcal{T}(\mathsf{Dec}, \hat{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}; \boldsymbol{r}))$.

PROOF. Let $A(x) = \hat{f}_{\boldsymbol{y},\boldsymbol{z},\boldsymbol{r}}(x)$. By perfect correctness, $f_{\boldsymbol{y},\boldsymbol{z}}(x) = \mathsf{Dec}(A(x))$ for every x. Hence, by Lemma 3.14, with overwhelming probability over $\boldsymbol{x}$, for every i we have

$$\partial_{x_i} f_{\boldsymbol{y},\boldsymbol{z}}(\boldsymbol{x}) = \mathcal{J}\mathsf{Dec}'(A(\boldsymbol{x})) \cdot \partial_{x_i} A'(\boldsymbol{x}),$$

where A' is the circuit outputted by $\mathcal{T}(A, \boldsymbol{x})$ and Dec' is the circuit outputted by $\mathcal{T}(\mathsf{Dec}, \boldsymbol{x})$. (The circuit A' is unknown to the attacker, who does not know $\boldsymbol{r}$, yet such a circuit exists.)

Fix some $i \in \{1, \ldots, n\}$. Observe that $A(x_1, \ldots, x_n) = (\hat{x}_1, \ldots, \hat{x}_n, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}, \hat{\boldsymbol{r}})$ and the only output that depends on x_i is $\hat{x}_i$ (since the encoding is decomposable). Hence, when deriving A' with respect to x_i, all the entries that do not correspond to the outputs $\hat{x}_i$ vanish. The previous equation therefore simplifies to

$$\partial_{x_i} f_{\boldsymbol{y},\boldsymbol{z}}(\boldsymbol{x}) = \mathcal{J}_{\hat{x}_i}\mathsf{Dec}'(A(\boldsymbol{x})) \cdot \partial_{x_i} A''(\boldsymbol{x}),$$

where A'' is the restriction of A' to the outputs that depend on $\hat{x}_i$. Recalling that $Y_i = \partial_{x_i} f_{\boldsymbol{y},\boldsymbol{z}}(\boldsymbol{x})$ completes the proof. $\square$

We move on to analyze the probability that $\mathcal{A}$ accepts a pseudorandom Y.

CLAIM 5.6. *It holds that*

$$\Pr_{\boldsymbol{y}_1 \xleftarrow{R} \{0,1\}^n}[\mathcal{A}(Y^1) = 1] = 1 - \mathrm{neg}(n),$$

where $Y^1 = \mathsf{PRG}_n(\boldsymbol{y}_1)$.

PROOF. Assume, toward a contradiction, that the claim does not hold and, for $\boldsymbol{y}_1 \xleftarrow{R} \{0, 1\}^n$, the adversary $\mathcal{A}$ accepts $Y^1 = \mathsf{PRG}(\boldsymbol{y}_1)$ with probability $1 - \varepsilon(n)$ for some nonnegligible ε. Consider the following adversary $\mathcal{B}(1^n)$ that attacks the computational privacy of randomized encodings:

Adversary $\mathcal{B}(1^n)$:

—Choose a prime $p > 2^{2s+1}$ for $s = \max \mathrm{size}\{f_n, \mathsf{Dec}_n \circ \hat{f}_n\}$.
—Let $\mathbb{F} = \mathrm{GF}(p)$ and send some standard implementation of $\mathbb{F}$ to the challenger.
—Sample $\boldsymbol{x}_0, \boldsymbol{z}_0 \xleftarrow{R} \mathbb{F}^n$, and let $\boldsymbol{y}_0 = 0^n$ and $Y^0 = \mathsf{PRG}(\boldsymbol{y}_0)$.

—Sample $\boldsymbol{x}_1 \xleftarrow{R} \mathbb{F}^n$, $\boldsymbol{y}_1 \xleftarrow{R} \{0,1\}^n$, and let $Y^1 = \mathsf{PRG}(\boldsymbol{y}_1)$ and $\boldsymbol{z}_1 = Y^0\boldsymbol{x}_0 + \boldsymbol{z}_0 - Y^1\boldsymbol{x}_1$.
—Send $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{z}_0)$ and $(\boldsymbol{x}_1, \boldsymbol{y}_1, \boldsymbol{z}_1)$ to the challenger.
—On input challenge $\boldsymbol{c}$, let $\mathsf{Dec}' := \mathcal{T}(\mathsf{Dec}, \boldsymbol{c})$.
—Output 1 if $Y_i^1 \in \mathsf{colSpan}(\mathcal{J}_{\hat{x}_i}\mathsf{Dec}'(\boldsymbol{c}))$ for all $i = 1, \ldots, n$. Otherwise, output 0.

Observe that f agrees on the 0-inputs $(\boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{z}_0)$ and on the 1-inputs $(\boldsymbol{x}_1, \boldsymbol{y}_1, \boldsymbol{z}_1)$, and so $\mathcal{B}$ is a valid adversary. When $b = 1$, by Claim 5.5, $\mathcal{B}$ outputs 1 with overwhelming probability. When $b = 0$, the probability that $\mathcal{B}$ returns 1 is exactly $1 - \varepsilon(n)$, since this is the probability that $\mathcal{A}$ accepts $Y_1 = \mathsf{PRG}(\boldsymbol{y}_1)$ for $\boldsymbol{y}_1 \xleftarrow{R} \{0,1\}^n$. (Indeed, observe that in both $\mathcal{A}$ and $\mathcal{B}$, the values $\boldsymbol{x}_1, \boldsymbol{z}_1$ are uniform over $\mathbb{F}^n$.) Overall the advantage of $\mathcal{B}$ is $\varepsilon(n)$, which, under our assumption, is nonnegligible, contradicting the computational privacy of the randomized encoding. □

We move on to show that if the encoding $\hat{x}$ is shorter than n^2, a random *binary string* $Y \in \{0,1\}^{n^2}$ is unlikely to be accepted by $\mathcal{A}$. As in the previous section, the proof uses a dimension argument.

CLAIM 5.7. *It holds that* $\Pr_{Y \xleftarrow{R} \{0,1\}^{n^2}}[\mathcal{A}(Y) = 1] \leq \frac{1}{2}$.

PROOF. Since $\hat{x}$ is shorter than n^2, there must be some $i \in \{1, \ldots, n\}$ for which $\hat{x}_i$ is shorter than n. We will focus on this i and show that for every fixed randomness $\boldsymbol{r}$ and for every $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{F}^n$, it holds that

$$\Pr_{Y \xleftarrow{R} \{0,1\}^{n^2}}\left[Y_i \in \mathsf{colSpan}\left(\mathcal{J}_{\hat{x}_i}\mathsf{Dec}'(\hat{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}; \boldsymbol{r}))\right)\right] \leq \frac{1}{2}. \tag{8}$$

Indeed, Dec has n outputs, and so $M := \mathcal{J}_{\hat{x}_i}\mathsf{Dec}'(\hat{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}; \boldsymbol{r}))$ is an $n \times |\hat{x}_i|$ matrix of elements in $\mathbb{F}$. Since $|\hat{x}_i| < n$, there exists a nontrivial vector $\boldsymbol{u} \in \mathbb{F}^n$ in the kernel of M, that is, $M \cdot \boldsymbol{u} = 0^{|\hat{x}_i|}$. Let j denote some nonzero coordinate of $\boldsymbol{u}$, namely, $\boldsymbol{u}_j \neq 0$. For every vector $\boldsymbol{v} \in \mathsf{colSpan}(M)$, it holds that $\boldsymbol{u}^{\mathrm{T}} \cdot \boldsymbol{v} = 0$. We will establish Equation (8) by showing that

$$\Pr_{Y_i \xleftarrow{R} \{0,1\}^n}[\boldsymbol{u}^{\mathrm{T}} \cdot Y_i = 0] \leq \frac{1}{2}.$$

To see this, observe that for any $w \in \{0,1\}^n$ that is orthogonal to $\boldsymbol{u}$, one can find a unique vector w' that is *not* orthogonal to $\boldsymbol{u}$ by letting $w' = w \oplus e_j$. (Here e_j denotes the jth unit vector, whose jth coordinate equals 1, and whose other coordinates are equal to 0.) Indeed, if $\boldsymbol{u}^{\mathrm{T}} \cdot w = 0$, then $\boldsymbol{u}^{\mathrm{T}} \cdot w' \in \{\pm \boldsymbol{u}_j\}$. The claim follows. □

Overall, $\mathcal{A}$ accepts random bitstrings with probability less than 1/2 and pseudorandom strings with probability $1 - \mathrm{neg}(n)$, and so it breaks the security of the PRG.

6. ARITHMETIC MULTIPARTY COMPUTATION

In this section, we prove lower bounds on the online communication of secure two-party computation in the arithmetic model.

6.1. Definitions

Secure computation with one-sided privacy. We consider a minimal form of Secure Computation with *one-sided privacy*. (This restriction only makes our lower bounds stronger.) Specifically, we will consider two-party functionalities in which only one party receives the output. Namely, Alice holds input x, Bob holds input y, and we want Alice to learn the functionality $f(x, y)$ without learning anything else. We require only one-sided security, and so Bob may learn everything. This notion of security is formalized later.

Recall that a two-party arithmetic protocol Π is a pair of (uniform) polynomial-size arithmetic circuits $A = \{A_n\}$ and $B = \{B_n\}$, which communicate with each other via special Send and $\mathsf{Receive}$ gates. Further recall that, for a field $\mathbb{F}$ and inputs $x, y \in \mathbb{F}^n$, the output of A is denoted by $\mathsf{Output}^{\Pi,\mathbb{F}_n}(x, y)$ and the view of A is denoted by $\mathsf{View}_A^{\Pi,\mathbb{F}}(x, y)$. The latter is the random variable that consists of the deterministic and random inputs of A_n and all the messages sent by B_n. (See Section 4.5 for formal definitions.)

Definition 6.1 (*Arithmetic Two-party Computation with One-sided Semihonest Security*). We say that a two-party arithmetic protocol $\Pi = (\{\mathsf{Alice}_n\}, \{\mathsf{Bob}_n\})$ *realizes* a (deterministic) functionality $f : \mathbb{F}^* \times \mathbb{F}^* \to \mathbb{F}^*$ with one-sided privacy if the following hold:

—Correctness: For every field $\mathbb{F}$ and inputs $x, y \in \mathbb{F}^n$,

$$\Pr\left[\mathsf{Output}^{\Pi,\mathbb{F}}(x, y) \neq f^{\mathbb{F}}(x, y)\right] = 0.$$

—One-sided privacy (against a semihonest Alice): For each polynomial-time Turing machine adversary $\mathcal{A}$, there exists a probabilistic polynomial-time Turing machine Sim such that the adversary $\mathcal{A}(1^n)$ can win the following game with probability at most $1/2 + neg(n)$:

—Given 1^n, the adversary $\mathcal{A}$ picks a field implementation $\mathbb{F}$, inputs $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}^{\text{poly}(n)}$, and outputs $\mathbb{F}, \boldsymbol{x}, \boldsymbol{y}$.

—The challenger samples $b \stackrel{R}{\leftarrow} \{0, 1\}$ and returns a view z where $z \stackrel{R}{\leftarrow} \mathsf{View}_A^{\Pi,\mathbb{F}}(\boldsymbol{x}, \boldsymbol{y})$ if $b = 0$, and $z \stackrel{R}{\leftarrow} \mathsf{Sim}(\boldsymbol{x}, f(\boldsymbol{x}, \boldsymbol{y}), \mathbb{F})$ if $b = 1$.

—The adversary $\mathcal{A}$ outputs a bit b' and wins if $b' = b$.

Note that the previous definition is *uniform* in the sense that the inputs x, y are chosen by a uniform adversary. Again, this only makes our lower bound stronger.

Clearly, there is a trivial protocol that achieves one-sided privacy: Alice sends her input x to Bob, who computes the function $f(x, y)$ and returns the result to Alice. In this protocol, Alice receives m field elements, where m is the output length of f. We will show that this is necessary even if the parties are allowed to exchange an arbitrary number of messages at an offline phase.

Online/Offline MPC. Without loss of generality, we assume that a protocol $\Pi = (\mathsf{Alice}, \mathsf{Bob})$ has the following format. The parties Alice and Bob use the randomness r_a, r_b, respectively. They first run the *offline* phase of the MPC protocol. We denote by $\hat{r}(r_a, r_b)$ the transcript of all the messages that Alice gets during the offline phase. After the offline phase, Alice receives an input x and Bob receives an input y. They exchange some further messages. We denote the messages *sent by* Bob as $\hat{t} = \hat{t}(r_a, r_b, x, y)$ and let $|\hat{t}|$ denote the *incoming online communication complexity* of Alice. (For simplicity, we assume that this amount depends only on the security parameter and not on the messages themselves.) At the end of the protocol, Alice runs a final algorithm on her view $(x, r_a, \hat{r}(r_a, r_b), \hat{t}(r_a, r_b, x, y))$ to get the output of the protocol. We refer to this final algorithm as the decoder, $\mathsf{Dec}(x, r_a, \hat{r}, \hat{t})$.

6.2. Main Result

We will prove that the online communication from Bob to Alice is as long as the output length of f. This lower bound applies even to the case where f is *strictly arithmetic*; that is, it does not contain division, $\mathsf{zerocheck}$ gates, or $\mathsf{bitsample}$ gates and simply computes a polynomial over $\mathbb{F}$.

Our lower bound relies on the existence of an *arithmetic pseudorandom generator* (see Definition 4.1). Recall that the output of an APRG is pseudorandom over $\mathbb{F}^{\ell}$ and its input (the seed) may contain both random field elements and random bits as long their total number is $n < \ell$. We will later (Section 9) show that APRG with polynomial stretch can be constructed based on the RLC assumption (with constant rate and constant noise rate). Furthermore, this APRG is *simple*; that is, it does not contain division or zerocheck gates (but uses random bits).

THEOREM 6.2. *Assume the existence of a simple arithmetic PRG. Then, for any arbitrary constant $c > 1$, there is a strictly arithmetic function $f : \mathbb{F} \times \mathbb{F}^n \to \mathbb{F}^{n^c}$ that cannot be realized with one-sided semihonest security by an arithmetic two-party protocol Π for which the incoming online communication complexity of Alice is shorter than $n^c - 1$ field elements.*

Remark 6.3 (*The Multiparty Case*). A similar bound on the communication applies in the multiparty setting even in the case of honest majority. Indeed, any t-party protocol Π for f in which the first two parties play the roles of Alice and Bob and all the other parties act as "servers" with no input or output immediately yields a two-party protocol Π' with one-sided semihonest security even if Π is only semihonest secure for coalitions of size 1.

PROOF OF THEOREM 6.2. Let $m = n^c$. We will later show (Lemma 9.2) that the existence of any (simple) APRG implies the existence of a (simple) APRG with seed length of n and output length of m. We view APRG as an arithmetic function $\mathsf{APRG} : \mathbb{F}^n \to \mathbb{F}^m$ by thinking of the random gates as inputs. This is done even for gates that sample random bits. We follow the convention that whenever a random *seed* $\boldsymbol{y}$ is chosen, it is chosen from the "right" distribution; that is, binary inputs are chosen as random $0-1$ values. As a result, $Y = \mathsf{APRG}(\boldsymbol{y})$ is pseudorandom over $\mathbb{F}^{2m}$. (Note however, that the pseudorandomness of the function APRG_n is not guaranteed for a y sampled from other distributions.)

We consider the function $f(x, y) = Y_0 x + Y_1$ where $(Y_0, Y_1) = \mathsf{APRG}(y)$. Alice's input to the function is x, while Bob holds y. Our proof shows that if APRG is a strictly arithmetic PRG, then we can break the computational privacy of any MPC protocol where the online communication ℓ from Bob to Alice is (strictly) shorter than $m-1$. We define the following adversary against the computational privacy of the MPC protocol. As in the previous section, we use subscripts to denote fixed inputs (e.g., $f_{\boldsymbol{y}}(x) = f(x, \boldsymbol{y})$).

Distinguisher $\mathcal{A}$:

- Let $\mathbb{F} = \mathrm{GF}(p)$ for a prime $p > 2 \cdot 2^{2s}$, where s is an upper bound on the accumulated size of all the circuits in Alice, Bob, and Dec and the circuit that computes f.
- Sample $\boldsymbol{x} \xleftarrow{R} \mathbb{F}$.
- Sample a random seed $\boldsymbol{y}$ for the APRG and compute $(Y_0, Y_1) := \mathsf{APRG}(\boldsymbol{y})$.
- Send to the challenger an implementation of $\mathbb{F}$ and the inputs $(\boldsymbol{x}, \boldsymbol{y})$.
- Given a challenge $(\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}})$, call the procedure $\mathsf{SpanCheck}(Y_0, (\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}}))$ defined as follows:
  —Let $B(x, \hat{t})$ denote the computation of the decoder with the values $\boldsymbol{r}_a$ and $\hat{\boldsymbol{r}}$ fixed, that is, $\mathsf{Dec}_{\boldsymbol{r}_a, \hat{\boldsymbol{r}}}(x, \hat{t})$.
  —Compute $B' = \mathcal{T}(B, (\boldsymbol{x}, \hat{\boldsymbol{t}}))$, where $\mathcal{T}$ is the zero-check removal procedure defined in Definition 3.13.
  —If $Y_0 \in \mathsf{colSpan}\mathcal{J}B'(\boldsymbol{x}, \hat{\boldsymbol{t}})$, return 0; otherwise, return 1.

In the following, we will show that if $(\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}})$ is sampled according to Alice's view (i.e., the challenge bit b is zero), then $\mathcal{A}$ returns 0 with overwhelming probability (Claim 6.4).

Table I. The Hybrid Distributions Over ($Y_0, (\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}})$)

	$\mathcal{D}_1$	$\mathcal{D}_2$	$\mathcal{D}_3$
	Sample random seed $\boldsymbol{y}$ $(Y_0, Y_1) = \mathsf{APRG}(\boldsymbol{y})$ $\boldsymbol{x} \xleftarrow{R} \mathbb{F}$ $(\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}}) \xleftarrow{R} \mathsf{Sim}(\boldsymbol{x}, Y_0\boldsymbol{x} + Y_1, \mathbb{F})$	$(Y_0, Y_1) \xleftarrow{R} \mathbb{F}^{2m}$	$(Y_0, Z) \xleftarrow{R} \mathbb{F}^{2m}$ $(\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}}) \xleftarrow{R} \mathsf{Sim}(\boldsymbol{x}, Z, \mathbb{F})$
Output:	$(Y_0, (\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}}))$		

In order to avoid cluttering, we only write modifications with respect to the previous distribution. For example, x in $\mathcal{D}_2$ is sampled as in $\mathcal{D}_1$; also, all three distributions output the tuple $(Y_0, (\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}}))$.

On the other hand, we will show (Claim 6.6) that if $(\boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}}) \xleftarrow{R} \mathsf{Sim}(\boldsymbol{x}, Y_0\boldsymbol{x} + Y_1, \mathbb{F})$ (i.e., the challenge bit b is one), $\mathcal{A}$ errs (outputs 0) with probability at most $1/2 + \text{neg}(n)$.

Claim 6.4. *For every fixed $\boldsymbol{y} \in \mathbb{F}^n$ and fixed randomness $\boldsymbol{r}_a$ and $\boldsymbol{r}_b$, it holds that*

$$\Pr_{\boldsymbol{x} \in \mathbb{F}}\left[Y_0 \in \mathsf{colSpan}\mathcal{J}B'(\boldsymbol{x}, \hat{\boldsymbol{t}})\right] \geq 1 - \text{neg}(n),$$

where Y_0 is the first half of the output of $\mathsf{APRG}(\boldsymbol{y})$*, $\hat{\boldsymbol{r}} = \hat{r}(\boldsymbol{r}_a, \boldsymbol{r}_b)$, $\hat{\boldsymbol{t}} = \hat{t}(\boldsymbol{r}_a, \boldsymbol{r}_b, \boldsymbol{x}, \boldsymbol{y})$, and B' is defined as in $\mathcal{A}$, that is, $B' = \mathcal{T}(\mathsf{Dec}_{\boldsymbol{r}_a, \hat{\boldsymbol{r}}}, (\boldsymbol{x}, \hat{\boldsymbol{t}}))$.*

Proof. Fix $\boldsymbol{r}_a, \boldsymbol{r}_b$, and $\boldsymbol{y}$, and consider the arithmetic circuit $A(x)$ that outputs $(x, \hat{t})$, where $\hat{t}$ denotes Bob's online messages that correspond to the inputs $(\boldsymbol{r}_a, \boldsymbol{r}_b, x, \boldsymbol{y})$. The circuit A can be obtained by composing the subcircuits of Alice and Bob, and therefore, A is of size smaller than s. From perfect correctness, we know that

$$f_{\boldsymbol{y}}(x) = B(A(x)),$$

where $f_{\boldsymbol{y}}(x) = f(x, \boldsymbol{y})$. Since the mapping APRG is strictly arithmetic, so is $f_{\boldsymbol{y}}(x)$. Therefore, we can apply Lemma 3.14 for the field $\mathbb{F}$ and get that

$$\Pr_{\boldsymbol{x} \xleftarrow{R} \mathbb{F}}\left[\partial_x f_{\boldsymbol{y}}(\boldsymbol{x}) = \mathcal{J}B'(A(\boldsymbol{x})) \cdot \partial_x A'(\boldsymbol{x})\right] \geq 1 - \text{neg}(n),$$

where $A' := \mathcal{T}(A, \boldsymbol{x})$. (Note that, by definition, $B' := \mathcal{T}(B, A(\boldsymbol{x}))$.)

Plugging in the definitions of A and f yields that, with overwhelming probability over $\boldsymbol{x}$,

$$Y_0 = \mathcal{J}B'(\boldsymbol{x}, \hat{\boldsymbol{t}}) \cdot \partial_x A'(\boldsymbol{x}),$$

and the claim follows. □

We showed that when $b = 0$, the adversary returns 0 and wins with overwhelming probability. Next, we prove that if $b = 1$, the adversary returns 0 with probability at most $\frac{1}{2} + \text{neg}(n)$. For this, we focus on the SpanCheck subroutine of $\mathcal{A}$. We analyze the acceptance probability of SpanCheck on three different (hybrid) distributions, where the first distribution corresponds to the distribution used by $\mathcal{A}$ when $b = 1$. The hybrid distributions are defined in Table I.

Claim 6.5. *For $(Y_0, \boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}}) \xleftarrow{R} \mathcal{D}_3$, it holds that*

$$\Pr\left[Y_0 \in \mathsf{colSpan}\mathcal{J}B'(\boldsymbol{x}, \hat{\boldsymbol{t}})\right] \leq \frac{1}{2},$$

where $B' = \mathcal{T}(\mathsf{Dec}_{\boldsymbol{r}_a, \hat{\boldsymbol{r}}}, (\boldsymbol{x}, \hat{\boldsymbol{t}}))$.

Proof. Fix $\boldsymbol{x}, Z$ and the coins of the simulator to some arbitrary values. Then, $M = \mathcal{J}B'(\boldsymbol{x}, \hat{\boldsymbol{t}})$ is an $m \times (\ell + 1)$ matrix over $\mathbb{F}$, and therefore its columns span a set S of at most $|\mathbb{F}|^{\ell+1}$ vectors in $\mathbb{F}^m$. Since Y_0 is distributed uniformly over $\mathbb{F}^m$ and independently of M, it falls in S with probability at most $\frac{|\mathbb{F}|^{\ell+1}}{|\mathbb{F}|^m}$, which is upper bounded by 1/2 when $\ell + 1 < m$. $\square$

CLAIM 6.6. *It holds that* $\mathcal{D}_1 \overset{c}{\approx} \mathcal{D}_3$.

PROOF. The proof proceeds via a hybrid argument. First, we claim that $\mathcal{D}_1 \overset{c}{\approx} \mathcal{D}_2$. Indeed, let $\mathcal{C}$ be a distinguisher between the two distributions; we construct an adversary $\mathcal{B}$ against the APRG. On input (Y_0, Y_1), the adversary $\mathcal{B}$ samples $\boldsymbol{x} \overset{R}{\leftarrow} \mathbb{F}$ and returns $\mathcal{C}(Y_0, \mathsf{Sim}(\boldsymbol{x}, Y_0\boldsymbol{x} + Y_1))$. If $\mathcal{B}$ receives as input a sample (Y_0, Y_1) from the APRG, then it passes $\mathcal{C}$ a random sample from $\mathcal{D}_1$. In turn, if $(Y_0, Y_1) \overset{R}{\leftarrow} \mathbb{F}^{2m}$, then it passes $\mathcal{C}$ a random sample from $\mathcal{D}_2$. Hence, $\mathcal{B}$ breaks the APRG with the same advantage that $\mathcal{C}$ distinguishes between the two distributions.

Next, it is not hard to verify that $\mathcal{D}_2$ is identically distributed to $\mathcal{D}_3$ as both Z and $Y_0\boldsymbol{x} + Y_1$ are uniformly distributed in $\mathbb{F}^m$. $\square$

By combining Claims 6.5 and 6.6, we conclude that for $(Y_0, \boldsymbol{x}, \boldsymbol{r}_a, \hat{\boldsymbol{r}}, \hat{\boldsymbol{t}}) \overset{R}{\leftarrow} \mathcal{D}_1$, it holds that

$$\Pr\left[Y_0 \in \mathsf{colSpan}\mathcal{J}B'(\boldsymbol{x}, \hat{\boldsymbol{t}})\right] \leq \frac{1}{2} + \mathrm{neg}(n),$$

where $B' = \mathcal{T}(\mathsf{Dec}_{\boldsymbol{r}_a, \hat{\boldsymbol{r}}}, (\boldsymbol{x}, \hat{\boldsymbol{t}}))$. This means that, when the challenge bit b equals 1, the adversary $\mathcal{A}$ outputs 1 with probability larger than $1/2 - \mathrm{neg}(n)$. Recall that, by Claim 6.4, if $b = 0$, the adversary $\mathcal{A}$ outputs 0 with probability $1 - \mathrm{neg}(n)$. Overall, it follows that $\mathcal{A}$ wins the game with a probability of at least $1/2 - \mathrm{neg}(n)$, thus breaking the computational privacy of the MPC protocol.

7. HOMOMORPHIC ENCRYPTION

In this section, we show that there are no homomorphic encryption schemes in the arithmetic model even for relatively simple operations like scalar multiplication and ciphertext addition. As explained in the introduction, our attacks rely on a reduction to the *Arithmetic Predictability Problem*. In Section 7.1, we will define this problem and show that it can be efficiently solved. (Some of the proofs will be deferred to Sections 7.4 and 7.5.) Homomorphic encryption will be defined and attacked in Sections 7.2 and 7.3.

7.1. Main Tool: Algorithm for the Arithmetic Predictability Problem

We define the Arithmetic Predictability Problem, denoted by $\mathsf{AP}_{d,\varepsilon,\Delta,p}$, which is parameterized by a degree bound $d \in \mathbb{N}$, proximity parameter $\varepsilon \in (0, 1)$, entropy gap $\Delta > 0$, and some prime p that defines the field $\mathbb{F} = \mathrm{GF}(p)$.

Definition 7.1 (*The Arithmetic Predictability Problem* $\mathsf{AP}_{d,\varepsilon,\Delta,p}$). Given an arithmetic circuit $P = (P_1, \ldots, P_m)$ with m outputs and n inputs $x = (x_1, \ldots, x_n)$, and an additional *target polynomial* $T(x_1, \ldots, x_m)$ (represented by an arithmetic circuit), distinguish between the following two cases:

—(**Yes case:**) There exists a (possibly inefficient) rational function $Q(z) = A(z)/B(z)$, where $A : \mathbb{F}^m \to \mathbb{F}$ and $B : \mathbb{F}^m \to \mathbb{F}$ are nonzero polynomials of degree at most d, such that

$$\Pr_{\boldsymbol{x} \overset{R}{\leftarrow} \mathbb{F}^n} [Q(P(\boldsymbol{x})) = T(\boldsymbol{x}) | B(P(\boldsymbol{x})) \neq 0] = 1. \tag{9}$$

—(**No case:**) There exists a probability distribution $(\mathbf{P}', \mathbf{T}')$ over $\mathbb{F}^m \times \mathbb{F}$ such that

$$(\mathbf{P}', \mathbf{T}') \quad \text{is } \varepsilon\text{-close to} \quad (P, T)(\boldsymbol{x}) \quad \text{where } \boldsymbol{x} \stackrel{R}{\leftarrow} \mathbb{F}^n \tag{10}$$

and

$$\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') \geq \log p - \Delta. \tag{11}$$

Recall that the *average min-entropy* $\tilde{\mathbf{H}}_\infty(Y|X)$ of a random variable Y given a random variable X measures (in logarithmic scale) the probability of guessing Y given X. (See Section 3.1.)

About the parameters. We briefly explain the role of the parameters. Consider, for starters, the case where Q is a *polynomial* (i.e., the denominator B is taken to be some nonzero scalar), and let us focus on a Yes instance. In this case, Q guesses $T(\mathbf{x})$ given $P(\mathbf{x})$, with probability 1 over $\mathbf{x} \stackrel{R}{\leftarrow} \mathbb{F}^n$. So $T(U_n)$ is *perfectly predictable* given $P(U_n)$, where U_n denotes the uniform distribution over $\mathbb{F}^n$. In the more general case, where the denominator of Q is a degree d polynomial, the predictor is allowed to "ignore" some of the inputs $\boldsymbol{x}$ (the ones that zero B). Hence, larger d weakens the predictability of Yes instances and allows us to consider richer predictors.

The parameters ε and Δ control the *unpredictability* of No instances. In the extreme case of $\varepsilon = \Delta = 0$, the random variable $T(U_n)$ is completely *unpredictable* given $P(U_n)$ and it cannot be guessed with probability better than $1/p$. Larger $\varepsilon, \Delta > 0$ weaken this condition and make No instances more predictable.[16]

A binary version of the previous problem (denoted by the *Gap-Learning Problem*) was studied by Applebaum et al. [2008] in the context of proving lower bounds for PAC learning. Indeed, thinking of $(P, T)(\mathbf{x})$ as a joint distribution over m-dimensional points $p \in \mathbb{F}^m$ and their label $t \in \mathbb{F}$, Yes instances are learnable (by a possibly inefficient learner), while No instances are information-theoretic unlearnable. The AP problem is also closely related to the problem of estimating the entropy of a given distribution (represented by a sampling circuit), which was studied by Goldreich and Vadhan [1999] for binary circuits and by Dvir et al. [2011] for arithmetic circuits. These binary variants are known to be complete for the class **SZK** of languages that admit statistical-zero knowledge proofs [Goldreich and Vadhan 1999; Applebaum et al. 2008], and so they are believed to be intractable.

Solving AP. We say that $\mathsf{AP}_{d,\varepsilon,p,\Delta}$ is efficiently solvable if given an input (P, T) of circuit size s it is possible to determine in time $\text{poly}(s, \log p)$ whether (P, T) is a Yes instance or a No instance with error probability of 1/3. We will show that, when $\mathbb{F}$ is sufficiently large, the problem can be solved efficiently. Recall that A is a *strictly* arithmetic circuit if it does not contain any $\mathsf{bitsample}$ gates, sample gates, $\mathsf{zerocheck}$ gates, or divide gates. (Recall that a generalized arithmetic circuit may use such gates.) We will always assume that the target polynomial T is a *strictly* arithmetic circuit as this will be the case in our attacks.

THEOREM 7.2. *The problem* $\mathsf{AP}_{d,\varepsilon,p,\Delta}$ *can be efficiently solved in the following cases:*

(*1*) *The inputs* $P : \mathbb{F}^n \to \mathbb{F}^m$ *and* $T : \mathbb{F}^n \to \mathbb{F}$ *are strictly arithmetic circuits that compute degree* d *polynomials,* $\varepsilon < \frac{1}{2}$, *and* $p > \frac{(n+1)d^{n+1}(1+2^\Delta)}{(1/2-\varepsilon)^2}$.

[16]We remark that the use of two parameters for unpredictability (i.e., ε and Δ), as opposed to a single unpredictability parameter, makes the notion more robust in a way that is crucial for our results.

(2) *The input* P *is an* s*-size (generalized) arithmetic circuit, the input* T *is an* s*-size strictly arithmetic circuit* $\varepsilon \leq 0.199$, *and* $p \geq (d+s)^{2s} \cdot 2^{3s+\Delta}$.[17]

In the second part, P is allowed to use internal random gates (which sample bits or field elements). In such a case, we assume that the probability distributions of Equations (9) and (10) are defined over the internal randomness of P, in addition to the random choice of $\boldsymbol{x} \xleftarrow{R} \mathbb{F}^n$.

Proof outline. The first part is proved via the following simple algorithm: choose a random point $\boldsymbol{x} \xleftarrow{R} \mathbb{F}^n$ and check if the rows of the Jacobian $\mathcal{J}P(\boldsymbol{x}) \in \mathbb{F}^{m\times n}$ span the gradient $\partial T(\boldsymbol{x}) \in \mathbb{F}^n$ of the target polynomial $T(x)$. The analysis follows the techniques of Dvir et al. [2009] and is deferred to Section 7.4. For the second part (which will be useful for our attack), we show how to map the input (P, T) into a strictly arithmetic circuit (P', T) via a sequence of (randomized) Karp reductions. The proof then follows by applying the first part of the theorem. (See Section 7.5.) We note that the second part of the theorem extends to the case where $\varepsilon \leq 1/5 - \alpha$ for any inverse polynomial α at the expense of letting $p = \Omega(\frac{\exp(2s^3+s\log(d+1)+\Delta)}{\alpha^2})$.

Remark 7.3. As we will later see, in order to attack arithmetic homomorphic encryption, it suffices to solve a very special case of $\mathsf{AP}_{d,\varepsilon,p,\Delta}$ where $\Delta = 0$, ε is negligible (or even taken to be zero), $d = 2^{\mathrm{poly}(s)}$, and p is a huge (exponentially large) prime. However, this should be done with respect to a generalized arithmetic circuit P that may use division gates, zero-testing gates, and bit-sample gates. The only way we know how to solve this problem is via a reduction to the strictly arithmetic case with a more general setting of parameters d, Δ, $\varepsilon > 0$.

7.2. Attacking Statistical Scalar-Multiplicative Homomorphic Encryption

Recall that an arithmetic encryption scheme is defined by a triple (KGen, Enc, Dec), where $\mathsf{KGen} = \{\mathsf{KGen}_n\}$, $\mathsf{Enc} = \{\mathsf{Enc}_n\}$, and $\mathsf{Dec} = \{\mathsf{Dec}_n\}$ are uniform sequences of polynomial-sized arithmetic circuits. (See Definition 4.2.) We assume that KGen_n outputs an additional *evaluation key* ek and that security holds even when the adversary is given ek at the first step of the security game. We further assume that the scheme satisfies the following notion of homomorphic evaluation (related notions have appeared in Ishai et al. [2005], Gentry [2009], and Bogdanov and Lee [2013]).

Definition 7.4 (*Scalar-multiplicative Homomorphic Evaluator*). An ε-statistical scalar-multiplicative homomorphic evaluator for an encryption scheme (KGen, Enc, Dec) is a uniform sequence of polynomial-sized (randomized) arithmetic circuits $\mathsf{sMul} = \{\mathsf{sMul}_n\}$ that takes as input a ciphertext c, a message (scalar) y, and an evaluation-key ek. It should satisfy the following conditions for all fields $\mathbb{F}$ and for all but a negligible fraction of the keys $(\mathsf{pk}, \mathsf{sk}, \mathsf{ek}) \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$:

—(Correctness) For every pair of messages $(x, y) \in \mathbb{F}^2$, with probability 1, the ciphertext $c = \mathsf{sMul}_n^{\mathbb{F}}(\mathsf{Enc}_n^{\mathbb{F}}(x, \mathsf{pk}), y, \mathsf{ek})$ decrypts correctly to $x \cdot y$, that is,

$$\Pr\left[\mathsf{Dec}_n^{\mathbb{F}}(c, \mathsf{sk}) = xy\right] = 1.$$

[17]Note that s implicitly upper-bounds the input and output lengths (n and m) and therefore we can state the result without putting an explicit bound on n and m.

—(Statistical homomorphism) For every pair of messages $x \in \mathbb{F}$ and $y \in \mathbb{F}$, the random variables

$$\left(c, \mathsf{sMul}_n^{\mathbb{F}}(c, y, \mathsf{ek})\right) \text{ and } \left(c, \mathsf{Enc}_n^{\mathbb{F}}(x \cdot y, \mathsf{pk})\right), \qquad \text{where } c \xleftarrow{R} \mathsf{Enc}_n^{\mathbb{F}}(x, \mathsf{pk}),$$

are $\varepsilon(n)$-close in statistical distance.

We say that sMul is an ε-*weak* scalar-multiplicative homomorphic evaluator if statistical homomorphism is replaced with the following property:

—(Weak homomorphism) For every pair of messages $x \in \mathbb{F}$ and $y \in \mathbb{F}$, the random variables

$$\mathsf{sMul}_n^{\mathbb{F}}\left(\mathsf{Enc}_n^{\mathbb{F}}(x, \mathsf{pk}), y, \mathsf{ek}\right) \quad \text{and} \quad \mathsf{Enc}_n^{\mathbb{F}}((x \cdot y), \mathsf{pk})$$

are $\varepsilon(n)$-close in statistical distance.

We rule out arithmetic encryption with statistical scalar-multiplicative homomorphism even in the weakest setting of symmetric encryption with one-time security. In this setting, we may assume, without loss of generality, that the scheme is perfectly correct.[18] In contrast, we will later show (Section 10) that weak multiplicative homomorphism can be achieved in the arithmetic model based on the RLC assumption.

THEOREM 7.5. *For every constant $\varepsilon \leq 0.199$, there is no one-time secure symmetric encryption scheme that is ε-statistical scalar multiplicative in the arithmetic model.*

PROOF. Let $s(n)$ be an upper bound on the circuit size of the homomorphic evaluator sMul_n and on the circuit size of the decryption algorithm. We will show that in order to attack the scheme, it suffices to solve the $\mathsf{AP}_{d,\varepsilon,p,\Delta}$ for $d = (s+1) \cdot 2^s$, $\Delta = 0$, $\varepsilon = 0.199$, and sufficiently large p. Specifically, the attacker $\mathcal{A}$ does the following:

—$\mathcal{A}$ receives 1^n, chooses a prime $p \in (2^{7s^2}, 2 \cdot 2^{7s^2})$, and sends some implementation of the field $\mathbb{F} = \mathrm{GF}(p)$ to the challenger.

—Given an evaluation key $\mathsf{ek} \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$, the adversary chooses a pair of messages $(x_0 = 0, x_1 = 1)$ and sends them to the challenger.

—Given a ciphertext c (which encrypts either x_0 or x_1), the adversary defines a randomized arithmetic circuit $P_{\mathsf{ek},c}(x) = \mathsf{sMul}_n^{\mathbb{F}}(c, x, \mathsf{ek})$ and a strictly arithmetic circuit $T(x) = x$. Then, it uses part 2 of Theorem 7.2 to classify (P, T) as a Yes instance or a No instance of $\mathsf{AP}_{d=(s+1)\cdot 2^s,\varepsilon,p,\Delta=0}$. If the answer is Yes, $\mathcal{A}$ outputs 1; otherwise, it outputs 0.

Fix the keys $(\mathsf{sk}, \mathsf{pk}, \mathsf{ek})$ that were chosen by $\mathsf{KGen}_n^{\mathbb{F}}$ and let us condition on the event that these keys are "good" in the sense that they satisfy the correctness and statistical homomorphism properties of Definition 7.4. (Recall that this event happens with all but negligible probability.) To analyze the success probability, it suffices to show that if c is an encryption of 1, then $(P_{\mathsf{ek},c}, T)$ is a Yes instance, and if c is an encryption of 0, then $(P_{\mathsf{ek},c}, T)$ is a No instance.

We begin with the case where c is an encryption of 1, and, correspondingly, $P_{\mathsf{ek},c}(x)$ outputs an encryption of $x \cdot 1 = x$. By perfect correctness, the decryption algorithm

[18]First, observe that without loss of generality, the decryption algorithm is deterministic (e.g., by letting the encryption output the randomness needed for decryption as part of the ciphertext). Next, note that any scheme with negligible decryption error of $\varepsilon(n)$ can be turned into a perfectly correct scheme. This is done by letting the encryption algorithm check if the resulting ciphertext decrypts correctly and, in case of an error, replace the ciphertext with the unencrypted message (together with a special "no encryption" flag). It is not hard to see that this transformation incurs only a negligible loss (of ε) in the security.

$\mathsf{Dec}_{\mathsf{sk}}(z) = \mathsf{Dec}(z, \mathsf{sk})$ satisfies $\mathsf{Dec}_{\mathsf{sk}}((P_{\mathsf{ek},c}(\boldsymbol{x})) = T(\boldsymbol{x})$ for every $\boldsymbol{x}$. By applying Proposition 3.12 to $\mathsf{Dec}_{\mathsf{sk}}$, there exists a rational function $\frac{A}{B}$ that is computed by a zerocheck-free circuit of size s and a degree $s2^s$-polynomial G such that

$$\frac{A}{B}(\boldsymbol{z}) = \mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{z}), \qquad \forall \boldsymbol{z} \text{ s.t } G(\boldsymbol{z}) \neq 0.$$

Consider the rational function $Q(z) = \frac{A(z)G(z)}{B(z)G(z)}$. This function certifies that $(P_{\mathsf{ek},c}, T)$ is a Yes instance, since $Q(P_{\mathsf{ek},c}(\boldsymbol{x})) = T(\boldsymbol{x})$ for every $\boldsymbol{x}$ that satisfies $B(P(\boldsymbol{x}))G(P(\boldsymbol{x})) \neq 0$. Finally, observe that the degree of the denominator and nominator of Q is upper bounded by $(s+1) \cdot 2^s$.

Next, we claim that if c is an encryption of 0, then $(P_{\mathsf{ek},c}, T)$ is a No instance. Indeed, in this case, $P_{\mathsf{ek},c}(x)$ outputs the ciphertext $x \cdot 0 = 0$, and, since the homomorphism is *statistical*, we have that for every good key pk and ek, and every message $\boldsymbol{x}$, the random variables

$$\big(c, \mathsf{sMul}_n^{\mathbb{F}}(c, \boldsymbol{x}, \mathsf{ek})\big) \qquad \text{and} \qquad \big(c, \mathsf{Enc}_n^{\mathbb{F}}(0, \mathsf{pk})\big)$$

are $\varepsilon(n)$-close. Letting $(\mathbf{P}', \mathbf{T}') = (\mathsf{Enc}_n^{\mathbb{F}}(0, \mathsf{pk}), T(\boldsymbol{x}))$ where $\boldsymbol{x} \xleftarrow{R} \mathbb{F}$, we conclude that

$$(P_{\mathsf{ek},c}(\boldsymbol{x}), T(\boldsymbol{x})) \quad \text{is } \varepsilon\text{-close to} \quad (\mathbf{P}', \mathbf{T}').$$

Since $\mathbf{T}' = T(\boldsymbol{x})$ is uniform and statistically independent of $\mathbf{P}' = \mathsf{Enc}_n^{\mathbb{F}}(0, \mathsf{pk})$, it follows that $\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') = \log p$, and so $(P_{\mathsf{ek},c}, T)$ is a No instance. The theorem follows. □

Remarks on Theorem 7.5

—For positive applications, one would typically strive for statistical homomorphic encryption with *negligible* error parameter ε. Theorem 7.5 therefore provides a strong negative result as it applies even to a constant $\varepsilon \leq 0.199$. (Similarly to Theorem 7.2, the lower bound holds for any value of $\varepsilon = \frac{1}{5} - \delta(n)$, where δ is an arbitrary inverse polynomial.)

—In the public-key setting, it is possible to extend the theorem to the case where the statistical homomorphism property (as defined in Definition 7.4) applies only to a noticeable fraction of the evaluation/public keys (ek, pk). Indeed, in this case, the attack $\mathcal{A}$ described in the proof of Theorem 7.5 succeeds over a noticeable fraction of the keys. The scheme can now be broken by applying $\mathcal{A}$ over "vulnerable" keys and using a random guess over all other keys (for which the statistical homomorphism does not hold). This strategy can be efficiently implemented since one can efficiently test (with high probability) whether the keys chosen by the challenger are vulnerable to $\mathcal{A}$ or not. (Just run the attack several times on a pair of "dummy ciphertexts" whose decryption is known and check whether the attack succeeds.) A similar extension holds for the case of CPA-secure *symmetric* encryption as in this case the encryption oracle can be used to test if the given evaluation key ek is vulnerable.

—Finally, we note that the attack generalizes to the case where the decryption algorithm can be written as a possibly inefficient polynomial of degree $2^{\text{poly}(n)}$. This extension can be used to handle restricted forms of arithmetic commitment schemes with statistical multiplicative homomorphism that support inefficient low-degree decryption.

7.3. Attacking Additive Homomorphic Encryption

Theorem 7.5 requires *statistical* homomorphism (for scalar multiplication); that is, it assumes that the homomorphic evaluator generates fresh ciphertexts that are *statistically close* to ciphertexts that were generated via the encryption algorithm. We move on to consider a weaker form of homomorphism (referred to as *multihop*) that only requires that repeated applications of the homomorphic evaluator preserve correctness. We extend our result to this more general setting at the expense of considering a more powerful family of homomorphic operations. Specifically, we consider encryptions that support homomorphic additions of ciphertexts, in addition to multiplications by (unencrypted) scalars.

Definition 7.6 (*Multihop Additive Homomorphic Encryption (AHE)*). An encryption scheme $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is multihop additively homomorphic if there exists a uniform sequence of polynomial-sized (randomized) arithmetic circuits $\mathsf{sMul} = \{\mathsf{sMul}_n\}$ and $\mathsf{Add} = \{\mathsf{Add}_n\}$ such that for all fields $\mathbb{F}$ and for all but a negligible fraction of the keys $(\mathsf{pk}, \mathsf{sk}, \mathsf{ek}) \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$, the following property holds:

—(Multihop) For every ciphertext c_1, c_2 that decrypts under $\mathsf{Dec}_n^{\mathbb{F}}(\cdot, \mathsf{sk})$ to $x_1, x_2 \in \mathbb{F}$ and for any scalar $y \in \mathbb{F}$, with probability 1, it holds that

$$\mathsf{Dec}_n^{\mathbb{F}}\big(\mathsf{Add}_n^{\mathbb{F}}(c_1, c_2, \mathsf{ek}), \mathsf{sk}\big) = x_1 + x_2 \quad \text{and} \quad \mathsf{Dec}_n^{\mathbb{F}}\big(\mathsf{sMul}_n^{\mathbb{F}}(c_1, y, \mathsf{ek}), \mathsf{sk}\big) = x_1 \cdot y.$$

Observe that we allow only multiplication by a pubic scalar and so AHE is far less powerful than fully homomorphic encryption. Moreover, we make no requirement regarding the distribution of homomorphically generated ciphertexts.

THEOREM 7.7. *There is no semantically secure multihop additively homomorphic symmetric encryption scheme in the arithmetic model.*

PROOF. Let $\ell(n)$ be an upper bound on the circuit size of the homomorphic addition and multiplication and on the circuit size of the decryption algorithm. The latter means that $\ell(n)$ also upper-bounds the length of a ciphertext. Let $s(n) = 3\ell^2(n)$. We will show that in order to attack the scheme, it suffices to solve the $\mathsf{AP}_{d,\varepsilon,p,\Delta}$ for $d = (s+1)\cdot 2^s$, $\Delta = 0$, $\varepsilon = \mathrm{neg}(n)$ and $p = O(2^{7s^2})$. It will be convenient to describe an attack against a multiple-message variant of the IND-CPA game where the challenge consists of two polynomially long vectors of messages. It is well known (cf. Katz and Lindell [2008, Chapter 3.4]) that this notion is equivalent to the single-message variant of IND-CPA (Definition 4.2), and it is not hard to verify that this equivalence carries over to the arithmetic setting.

We move on to describe the attacker $\mathcal{A}$. In the following, we use $\boxplus$ to denote homomorphic addition and $\boxtimes$ to denote homomorphic multiplication of a ciphertext by an unencrypted scalar (and omit for readability the dependency in the security parameter, in the field and in the evaluation key):

—$\mathcal{A}$ receives 1^n, chooses a prime $p \in (2^{7s^2}, 2\cdot 2^{7s^2})$, and sends some implementation of the field $\mathbb{F} = \mathrm{GF}(p)$ to the challenger.

—Given an evaluation key $\mathsf{ek} \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$, the adversary sends to the challenger a pair (y, w) of message vectors, where $y \xleftarrow{R} \mathbb{F}^{t+1}$ consists of $t+1$ random messages $(y_0, \ldots, y_t)$ and $w = 0^{t+1}$ is the all-zero vector where $t = \ell + 2$.

—Given a vector of ciphertexts $c = (c_0, \ldots, c_t)$ (which encrypts either y or w), the adversary defines a (generalized) arithmetic circuit $P_{\mathsf{ek},c} : \mathbb{F}^t \to \mathbb{F}^\ell$ over the (formal) inputs $x = (x_1, \ldots, x_t)$ as follows:

$$P_{\mathsf{ek},c}(x) = c_0 \boxplus (c_1 \boxtimes x_1) \boxplus \cdots \boxplus (c_t \boxtimes x_t);$$

that is, P homomorphically computes an "inner product" between (the plaintexts that correspond to) c and the plaintext vector $(1, x)$. The adversary also defines a strictly arithmetic single-output circuit:

$$T_y(x) = y_0 + (y_1 x_1) + \cdots + (y_t x_t).$$

Since both circuits are of size at most s, the adversary can use part 2 of Theorem 7.2 to classify (P, T) as a Yes instance or a No instance of $\mathsf{AP}_{d,\varepsilon,p,\Delta}$. If the answer is Yes, $\mathcal{A}$ outputs 1; otherwise, it outputs 0.

As in the proof of Theorem 7.5, we fix the keys $(\mathsf{sk}, \mathsf{pk}, \mathsf{ek})$ that were chosen by $\mathsf{KGen}_n^{\mathbb{F}}$ and condition on the event that these keys are "good" in the sense that they satisfy the Correctness property of Definition 7.6. Observe that when c is an encryption of the vector y, the circuit $P_{\mathsf{ek},c}(x)$ outputs an encryption of the plaintext outputted by the target circuit $T_y(x)$. Using the same argument as in Theorem 7.5, it follows that, in this case, $(P_{\mathsf{ek},c}, T_y)$ is a Yes instance. To conclude the proof, it suffices to show that if c is an encryption of 0^{t+1}, then $(P_{\mathsf{ek},c}, T_y)$ is, with probability $1 - o(1)$ over the choice of y, a No instance.

Consider the jointly distributed random variables:

$$(\boldsymbol{y}, P_{\mathsf{ek},c}(\boldsymbol{x}), T_{\boldsymbol{y}}(\boldsymbol{x})), \quad \text{where } \boldsymbol{y} \xleftarrow{R} \mathbb{F}^{t+1}, \boldsymbol{x} \xleftarrow{R} \mathbb{F}^t. \tag{12}$$

First, observe that $P_{\mathsf{ek},c}(\boldsymbol{x})$ consists of only $\ell(n)$ entries. Hence, by Fact 3.1 item 3, the unpredictability of $\boldsymbol{x}$ conditioned on $P_{\mathsf{ek},c}(\boldsymbol{x})$ is at least

$$\tilde{\mathbf{H}}_\infty(\boldsymbol{x} | P_{\mathsf{ek},c}(\boldsymbol{x})) \geq (t - \ell) \log p = 2 \log p.$$

Next, observe that since c is an encryption of zeroes, $P_{\mathsf{ek},c}(\boldsymbol{x})$ is statistically independent of $\boldsymbol{y}$. Finally, note that $T_{\boldsymbol{y}}(\boldsymbol{x})$ computes a pairwise independent hash function from $\mathbb{F}^t$ to $\mathbb{F}$. We can therefore think of Equation (12) as composed of a key $\boldsymbol{y} \xleftarrow{R} \mathbb{F}^{t+1}$, some "leakage" $I = P_{\mathsf{ek},c}(\boldsymbol{x})$ on the source $\boldsymbol{x} \xleftarrow{R} \mathbb{F}^t$, and the value of the hash function $T_{\boldsymbol{y}}$ applied to the source $\boldsymbol{x}$. By the generalized hashing lemma (Fact 3.2), the triple (12) is $\frac{1}{2\sqrt{p}}$-close in statistical distance to the triple

$$(\boldsymbol{y}, P_{\mathsf{ek},c}(\boldsymbol{x}), \mathbf{T}'), \quad \text{where } \mathbf{T}' \xleftarrow{R} \mathbb{F}. \tag{13}$$

By Markov's inequality, it follows that, for all but $1/n$ fraction of possible $\boldsymbol{y}$s, it holds that

$$\Delta((P_{\mathsf{ek},c}(\boldsymbol{x}), T_{\boldsymbol{y}}(\boldsymbol{x})), (P_{\mathsf{ek},c}(\boldsymbol{x}), \mathbf{T}')) \leq \frac{n}{2\sqrt{p}} \leq \mathrm{neg}(n).$$

This shows that $(P_{\mathsf{ek},c}, T_{\boldsymbol{y}})$ is a No instance, and the theorem follows. □

Remark 7.8. It is not hard to verify that the proof of Theorem 7.7 rules out the existence of any homomorphic encryption that allows one to *compactly* evaluate *inner products*. The latter means that for some polynomial $t = t(n)$, the homomorphic evaluator can map a vector of plaintexts $y = (y_1, \ldots, y_t) \in \mathbb{F}^t$ and a vector of ciphertexts $(c_1, \ldots, c_t)$ that encrypts the plaintexts $(x_1, \ldots, x_t) \in \mathbb{F}^t$ to a new ciphertext c^* that decrypts to $\sum x_i y_i$ and consists of less than $t - 3$ field elements.[19]

7.4. Proof of Theorem 7.2: Part 1

The idea is to output Yes if the rows of the Jacobian $\mathcal{J}P : \mathbb{F}^n \to \mathbb{F}^{m \times n}$ span the gradient $\partial T(x) : \mathbb{F}^n \to \mathbb{F}^n$ of the target polynomial $T(x)$, where the underlying field is the field of

[19]Note that without any compactness constraint, homomorphism becomes trivial (and useless) as the "homomorphic evaluator" can simply output the given ciphertexts and leave the actual computation to the decryptor.

rational functions $\mathbb{F}(x)$ and $\partial T(x)$ is viewed as a row vector. As observed by Dvir et al. [2009] (see also Kayal [2009, Corollary 3]), this can be checked efficiently (with small error) via the following randomized algorithm.

The algorithm $\mathcal{A}$. Choose a random vector $\mathbf{x} \overset{R}{\leftarrow} \mathbb{F}^n$ and output "Yes" if and only if the rows of the matrix $\mathcal{J}P(\mathbf{x}) \in \mathbb{F}^{m\times n}$ span (over $\mathbb{F}$) the row vector $\partial T(\mathbf{x}) \in \mathbb{F}^n$.

Let us analyze the algorithm $\mathcal{A}$ beginning with the case of a Yes instance.

CLAIM 7.9. *If the input is a Yes instance, then the algorithm outputs Yes with probability* $1 - \frac{2d^2}{p} > \frac{2}{3}$.

PROOF. Assume that the input is a Yes instance. Namely, for some m-variate rational function $Q(z_1, \ldots, z_m) = A(z_1, \ldots, z_m)/B(z_1, \ldots, z_m)$, it holds that

$$Q(P(\boldsymbol{x})) = T(\boldsymbol{x}),$$

for every $\boldsymbol{x}$ that does not zero $B(P(\boldsymbol{x}))$. Note that the degrees of the denominator and nominator of the LHS are upper bounded by d^2 and the degree of the polynomial of the RHS is upper bounded by d. Since $\mathbb{F}$ is sufficiently large, it follows (by the Schwartz-Zippel lemma) that the rational functions $Q(P(x))$ and $T(x)$ are equivalent. Therefore, by the chain rule (Lemma 3.7), we have that

$$\partial_z Q(P(x)) \cdot \mathcal{J}_x P(x) \equiv \partial_x T(x),$$

where $\partial_z Q(P(x))$ and $\partial_x T(x)$ are treated as row vectors. Hence, if the vector $\mathbf{x} \in \mathbb{F}^n$ does not zero the denominator of $\partial_z Q(P(x))$, the test will pass and the algorithm will output "Yes."

We show that the denominator of $\partial_z Q(P(x))$ is zeroed with probability at most $2d^2/|\mathbb{F}|$. Recall that $Q(z) = A(z)/B(z)$ and so the denominator of $\partial_z Q(P(x))$ is $B^2(P(x))$, which is a polynomial of degree at most $2d^2$ (since d upper-bounds the degree of B and the degree of the polynomials P). Hence, by the Schwartz-Zippel lemma, a random $\mathbf{x} \overset{R}{\leftarrow} \mathbb{F}^n$ will zero $\partial_z Q(P(x))$ with probability at most $2d^2/|\mathbb{F}|$, which, for our choice of parameters, is (much) smaller than 1/3. □

We move on to the case of a No instance. We will need the following fact from Dvir et al. [2009], originally proved over the field of rational numbers by Ehrenborg and Rota [1993].

FACT 7.10 (THEOREMS 3.1 AND 3.3 IN DVIR ET AL. [2009]). *Let $\mathbb{F} = \mathrm{GF}(p)$ and let $M = (M_1, \ldots, M_m)$ be an m-tuple of degree d polynomials in $\mathbb{F}[x_1, \ldots, x_n]$, where p is a prime larger than $D \overset{def}{=} (n+1)d^n$. Then, the rows of the Jacobian $\mathcal{J}M$ are linearly dependent (over the field of rational functions) if and only if there exists a polynomial $Z \in \mathbb{F}[z_1, \ldots, z_m]$ such that $Z(M_1(x), \ldots, M_m(x)) \equiv 0$. Furthermore, in the "only if" direction, Z is guaranteed to be of degree at most D.*

We remark that the "if" direction holds even if p is small. The polynomial Z is referred to as the *annihilating polynomial* of M and its existence corresponds to the classical notion of *algebraic dependence*.

We begin by showing that, in the case of a No instance, the Jacobian $\mathcal{J}P$ does not span the gradient ∂T of the target polynomial over the field of rational functions $\mathbb{F}(x)$.

CLAIM 7.11. *If the input is a No instance, then* $\partial T \notin \mathrm{span}(\mathcal{J}P)$.

PROOF. Assume toward a contradiction that $\partial T \in \mathrm{span}(\mathcal{J}P)$. Without loss of generality, we assume that the rows of $\mathcal{J}P$ are linearly independent. (Otherwise, take some minimal subset P' of the polynomials in P whose Jacobian spans $\mathcal{J}P$; observe that if (P, T) is a No instance, then so is (P', T) for any subset of the polynomials $P' \subseteq P$.)

By Fact 7.10, there exists an annealing polynomial $Z(z_1, \ldots, z_m, t)$ of degree $D = (n+1)d^n$ such that $Z(T, P) \equiv 0$. Furthermore, since the rows of $\mathcal{J}P$ are linearly independent, by Fact 7.10, the polynomials P have no annealing polynomial. It follows that Z can be written as

$$Z(z,t) = \sum_{i=1}^{D} t^i \cdot Z_i(z), \qquad \text{where the polynomial } Z_i(P(x)) \not\equiv 0 \text{ for some } i. \tag{14}$$

Since (P, T) is assumed to be a No instance, there exists a distribution $(\mathbf{P}', \mathbf{T}')$ such that (1) $(\mathbf{P}', \mathbf{T}')$ is ε-close to $(P, T)(U_n)$ and (2) $\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') \geq \log p - \Delta$. We will use (2) to derive a contradiction to (1). That is, we describe an (inefficient) distinguisher $\mathcal{A}$ for $(\mathbf{P}', \mathbf{T}')$ and $(P, T)(U_n)$.

Given a vector $(z = (\mathbf{z}_1, \ldots, \mathbf{z}_m), \mathbf{t}) \in \mathbb{F}^{m+1}$, our distinguisher $\mathcal{A}$ accepts the input if and only if $Z((\mathbf{z}_1, \ldots, \mathbf{z}_m), \mathbf{t}) = 0$. By definition, the test accepts every value (z, t) in the image of $(P, T)(x)$; therefore,

$$\Pr_{\mathbf{x}}[\mathcal{A}(P(\mathbf{x}), T(\mathbf{x})) = 1] = 1.$$

We analyze the acceptance probability of $(\mathbf{P}', \mathbf{T}')$. For a fixed $\boldsymbol{z} \in \mathbb{F}^m$, let $L_{\boldsymbol{z}}$ denote the set of $t \in \mathbb{F}$ for which the distinguisher accepts the value $(\boldsymbol{z}, t)$. Namely,

$$L_{\boldsymbol{z}} \stackrel{\text{def}}{=} \{t \in \mathbb{F} : Z(\boldsymbol{z}, t) = 1\}.$$

First, observe that

$$\Pr_{\boldsymbol{z} \xleftarrow{R} P(U_n)}[|L_{\boldsymbol{z}}| \leq Dd] > 1 - \frac{Dd}{p}. \tag{15}$$

Indeed, let i be the maximal integer for which $Z_i(P(x))$ is a nonzero polynomial, as promised by Equation (14). As a polynomial in x, the degree of $Z_i(P(x))$ is at most $(D - i) \cdot d < Dd$; hence, by the Schwartz-Zippel lemma, $\Pr_{\mathbf{x}}[Z_i(P(\mathbf{x})) = 0] < Dd/|\mathbb{F}|$. For any fixed $\mathbf{x}$ for which $Z_i(P(\mathbf{x})) \neq 0$, the residual polynomial $Z_{\boldsymbol{x}}(t) = Z(P(\mathbf{x}), t)$ is a nonzero polynomial of degree at most $i \cdot d < Dd$ and so it has at most Dd roots and Equation (15) follows.

Since the marginals $\mathbf{P}'$ and $P(U_n)$ are ε-close, Equation (15) implies that

$$\Pr_{\boldsymbol{z} \xleftarrow{R} \mathbf{P}'}[|L_z| \leq Dd] > 1 - \frac{Dd}{p} - \varepsilon. \tag{16}$$

Recall that $\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') \geq \log p - \Delta$, and so by Markov's inequality (Fact 3.1, item 1), we have that for every $\delta > 0$,

$$\Pr_{\boldsymbol{z} \xleftarrow{R} \mathbf{P}'}[\mathbf{H}_\infty(\mathbf{T}'|\mathbf{P}' = \boldsymbol{z}) \geq \log p - \Delta - \log(1/\delta)] \geq 1 - \delta. \tag{17}$$

Call $\boldsymbol{z}$ *good* if both $|L_z| \leq Dd$ and $\mathbf{H}_\infty(\mathbf{T}'|\mathbf{P}' = \boldsymbol{z}) \geq \log p - \Delta - \log(1/\delta)$. Combining Equations (16) and (17) and applying a union bound, it follows that a random $\boldsymbol{z} \xleftarrow{R} \mathbf{P}'$ is good with probability $1 - \delta - \varepsilon - \frac{Dd}{p}$. Finally, observe that for a good $\boldsymbol{z}$, we have that

$$\Pr[\mathbf{T}' \in L_{\boldsymbol{z}}|\mathbf{P}' = \boldsymbol{z}] \leq Dd \cdot 2^{-\log p + \Delta + \log(1/\delta)} \leq \frac{Dd2^\Delta}{\delta p}.$$

Hence, the distinguisher $\mathcal{A}$ accepts $(\mathbf{P}', \mathbf{T}')$ with probability at most $(\delta + \frac{Dd}{p} + \varepsilon) + \frac{Dd2^{\Delta}}{\delta p}$. Overall, the distinguishing advantage is $1 - (\delta + \frac{Dd}{p} + \varepsilon + \frac{Dd2^{\Delta}}{\delta p})$. Taking $\delta = (1-2\varepsilon)/2$ and plugging in $p > \frac{Dd(1+2^{\Delta})}{(1/2-\varepsilon)^2}$, we obtain a distinguishing advantage larger than ε. □

Finally, it remains to show that if $\partial T \notin \mathrm{span}(\mathcal{J}P)$, then for a random $\mathbf{x} \xleftarrow{R} \mathbb{F}^n$, the matrix $\partial T(\mathbf{x}) \notin \mathrm{span}(\mathcal{J}P)(\mathbf{x})$. This follows from the following standard claim.

CLAIM 7.12. *Let M be an $m \times n$ matrix of degree d polynomials in $\mathbb{F}[x_1, \ldots, x_n]$ and let v be a vector of n polynomials of degree d over $\mathbb{F}[x_1, \ldots, x_n]$. Then, if the rows of the Jacobian M do not span v over the field of rational functions $\mathbb{F}(x)$, then* $\Pr_{\mathbf{x}}[v(\mathbf{x}) \in \mathrm{span}(M(\mathbf{x}))] \leq dn/|\mathbb{F}|$.

PROOF. Let M' be the matrix obtained by taking a subset of the rows of M that forms a basis for M. Since the matrix M' has full rank and since it does not span v, the number of rows $m' \leq m$ of M' is strictly smaller than the number of columns n. We define an $m' \times (m'+1)$ matrix M'' by removing an arbitrary set of $n - m' - 1$ columns from M' and define v' by removing the corresponding entries of v. Observe that

$$\Pr_{\mathbf{x}}[v(\mathbf{x}) \in \mathrm{span}(M(\mathbf{x}))] = \Pr_{\mathbf{x}}[v(\mathbf{x}) \in \mathrm{span}(M'(\mathbf{x}))] \leq \Pr_{\mathbf{x}}[v'(\mathbf{x}) \in \mathrm{span}(M''(\mathbf{x}))].$$

Hence, it suffices to show that the square matrix $T = \binom{M''}{v}$ of polynomials is likely to be nondegenerate when evaluated on a random point $\mathbf{x}$. Indeed,

$$\Pr_{\mathbf{x}}[\det(T(\mathbf{x})) = 0] = \Pr_{\mathbf{x}}[\det(T)(\mathbf{x}) = 0] \leq d^n/|\mathbb{F}|,$$

where $\det(T)$ is the polynomial $P(x)$ that corresponds to the determinant of T computed over the field of rational functions. The last inequality follows by Schwartz-Zippel, noting that P is a nonzero polynomial (since T is nondegenerate over $\mathbb{F}(x)$) of degree at most nd. □

Combining the last two claims, we conclude that No instances are accepted with probability at most 1/3. This completes the proof of Theorem 7.2. □

7.5. Proof of Theorem 7.2: Part 2

The proof follows by a sequence of Karp reductions from generalized Arithmetic Circuits to strictly arithmetic circuits.

7.5.1. Removing Random Gates. We begin by describing how to remove sample gates (which sample uniform field elements) and bitsample gates (which sample uniform 0-1 values). Handling the first type is easy: just view the sample gates as additional input gates. Formally, let $\ell \leq s$ denote the number of sample gates, and for $x \in \mathbb{F}^n$ and $R \in \mathbb{F}^{\ell}$, let $P(x; R)$ denote the outcome of $P(x)$ when the sample gates are fixed to the value R. Then, the circuit $P'(x, R) = P(x; R)$ has no sample gates. Furthermore, it is not hard to verify that if (P, T) is a Yes instance (No instance, respectively), then so is (P', T'), where $T'(x, R) = T(x)$.

The case of bitsample gates is slightly more complicated. Let $T : \mathbb{F}^n \to \mathbb{F}$ be an s-size strictly arithmetic circuit and let $P : \mathbb{F}^n \to \mathbb{F}^m$ be an s-size arithmetic circuit that uses $\ell < s$ bitsample gates. For a string $\rho \in \{0,1\}^{\ell}$, let P_ρ denote the arithmetic circuit obtained by fixing the value of the bitsample gates of P to ρ.

LEMMA 7.13 (REMOVING bitsample GATES). *For every $\alpha > 0$, the following hold:*

—*If (P, T) is a Yes instance of* $\mathsf{AP}_{d,\varepsilon,p,\Delta}$*, then, for every ρ, (P_ρ, T) is a Yes instance of* $\mathsf{AP}_{d,\varepsilon',p,\Delta'}$.

—*If* (P, T) *is a No instance of* $\mathsf{AP}_{d,\varepsilon,p,\Delta}$*, then, with probability* α *over* $\rho \xleftarrow{R} \{0,1\}^\ell$*, the pair* (P_ρ, T) *is a No instance of* $\mathsf{AP}_{d,\varepsilon',p,\Delta'}$*,*

where $\varepsilon' = \frac{2\varepsilon}{1-\varepsilon-3\alpha/2}$ *and* $\Delta' = \Delta + s + \log(1/\alpha) + 1$.

Proof. The first item holds by definition. (Indeed, (P_ρ, T) is a Yes instance of $\mathsf{AP}_{d,\varepsilon,p,\Delta}$, and therefore also a Yes instance of $\mathsf{AP}_{d,\varepsilon',p,\Delta'}$.) We will prove that, with high probability, a No instance is mapped to a No instance. Let R be the uniform distribution over $\{0,1\}^\ell$ strings. Since (P, T) is a No instance, there exists a distribution $(\mathbf{P}', \mathbf{T}')$ that is ε-close to $(P(U_n, R), T(U_n))$ and for which $\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') > \log p - \Delta$. Consider the joint distribution $(P(U_n, R), T(U_n), R)$ and observe that we can define a new random variable $\mathbf{R}'$ that is jointly distributed with $(\mathbf{T}', \mathbf{P}')$ such that the distributions

$$(P(U_n, R), T(U_n), R) \quad \text{and} \quad (\mathbf{P}', \mathbf{T}', \mathbf{R}') \quad \text{are } \varepsilon\text{-close.} \tag{18}$$

Specifically, let $\mathbf{R}' = f(\mathbf{P}', \mathbf{T}')$, where $f(p,t)$ is the randomized mapping that given $(p,t) \in \mathbb{F}^m \times \mathbb{F}$ uniformly samples (r, x) subject to $P(x,r) = p$ and $T(x) = t$ and outputs r. Observe that

$$(P(U_n, R), T(U_n), R) \overset{i}{\equiv} (P(U_n, R), T(U_n), f(P(U_n, R), T(U_n)))$$

(recall that $X \overset{i}{\equiv} Y$ means that X and Y are identically distributed). Also, by definition,

$$(\mathbf{T}', \mathbf{P}', \mathbf{R}') \overset{i}{\equiv} (\mathbf{T}', \mathbf{P}', f(\mathbf{P}', \mathbf{T}')).$$

Hence, Equation (18) follows from the fact that $(\mathbf{P}, \mathbf{T})$ and $(P(U_n, R), T(U_n))$ are ε-close.

We will use the following notation. For a jointly distributed (X, Y), let $X_{Y=y}$ denote the distribution of X conditioned on the event $Y = y$. Let $\alpha_1, \alpha_2 > 0$. Call $\rho \in \{0,1\}^\ell$ *good* if it satisfies the following conditions (Equations (19) and (20)):

$$(P(U_n, R), T(U_n))_{R=\rho} \quad \text{and} \quad (\mathbf{T}', \mathbf{P}')_{\mathbf{R}'=\rho} \quad \text{are } (2\varepsilon/\alpha_1) - \text{close,} \tag{19}$$

and

$$\tilde{\mathbf{H}}_\infty(\mathbf{T}'_{\mathbf{R}'=\rho} | \mathbf{P}'_{\mathbf{R}'=\rho}) > \log p - \Delta - s - \log(1/\alpha_2). \tag{20}$$

Note that if ρ is good, then (P_ρ, T) is indeed a No instance of $\mathsf{AP}_{d,\varepsilon',p,\Delta'}$. We begin by showing that a large fraction of $\rho \in \{0,1\}^\ell$ satisfies Equation (19). We will need the following simple fact.

Fact 7.14. *If* (X, Y) *is* ε*-close to* (X', Y')*, then* $\Pr_{y \xleftarrow{R} Y}[\Delta(X_y, X'_y) > 2\varepsilon/\alpha] < \alpha$*, where* X_y *(*X'_y*, respectively) denotes the distribution of* X *(*X'*, respectively) conditioned on the event* $Y = y$ *(*$Y' = y$*, respectively).*

Proof. By Markov's inequality, it suffices to prove that

$$\mathop{\mathbb{E}}_{y \xleftarrow{R} Y} [\Delta(X_y, X'_y)] \le 2\varepsilon.$$

For every y and distinguisher A, let $\alpha(y, A) = \Pr[A(X_y)] - \Pr[A(X'_y)]$, where $\Pr[A(x)]$ abbreviates $\Pr[A(x) = 1]$. Let A_y be a distinguisher that maximizes, over all distinguishers A, the quantity $\alpha(y, A)$, and let $\alpha_y = \alpha(y, A)$. Note that α_y is the statistical distance between X_y and X'_y. We can write

$$\begin{aligned}
\mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y}[\Pr[A_y(X_y)] - \Pr[A_y(X'_y)]] &= \mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y}[\Pr[A_y(X_y)]] - \mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y}[\Pr[A_y(X'_y)]] \\
&= \mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y}[\Pr[A_y(X_y)]] - \left(\mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y'}[\Pr[A_y(X'_y)]] - \mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y'}[\Pr[A_y(X'_y)]]\right) \\
&\quad - \mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y}[\Pr[A_y(X'_y)]] \\
&= \left(\mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y}[\Pr[A_y(X_y)]] - \mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y'}[\Pr[A_y(X'_y)]]\right) \\
&\quad + \left(\mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y'}[\Pr[A_y(X'_y)]] - \mathop{\mathbb{E}}_{y \stackrel{R}{\leftarrow} Y}[\Pr[A_y(X'_y)]]\right).
\end{aligned}$$

Letting $A(x, y) = A_y(x)$, we can rewrite the first difference as $\Pr[A(X, Y)] - [\Pr[A(X', Y')]$ and so it is upper bounded by ε. The second difference can be written as

$$\Pr[B(Y')] - \Pr[B(Y)],$$

where $B(y)$ is the randomized function that samples $x \stackrel{R}{\leftarrow} X'_y$ and outputs $A_y(x)$. Since the marginals Y and Y' are also ε-close, it follows that the second difference is also upper bounded by ε, which completes the proof. □

By Fact 7.14 and Equation (18), a random $\rho \stackrel{R}{\leftarrow} R$ satisfies Equation (19) with probability $1 - \alpha_1$. We show that a random $\rho \stackrel{R}{\leftarrow} R$ satisfies Equation (20) with probability $1 - \alpha_2$.

Since $\mathbf{R}'$ is of bit-length $\ell < s$, we have, by Fact 3.1 [Dodis et al. 2008, Lemma 2.2b],

$$\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}', \mathbf{R}') > \tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') - s > \log p - \Delta - s;$$

it follows (by the second part of Fact 3.1) that

$$\Pr_{\rho \stackrel{R}{\leftarrow} \mathbf{R}'}[\tilde{\mathbf{H}}_\infty(\mathbf{T}'_{\mathbf{R}'=\rho}|\mathbf{P}'_{\mathbf{R}'=\rho}) > \log p - \Delta - s - \log(1/\alpha_2)] > 1 - \alpha_2.$$

Since $\mathbf{R}'$ is ε-close to R, we conclude that a random $\rho \stackrel{R}{\leftarrow} R$ satisfies Equation (20) with probability $1 - \alpha_2 - \varepsilon$. Overall, by a union bound, a random $\rho \stackrel{R}{\leftarrow} R$ is likely to be good with probability $1 - \alpha_1 - \alpha_2 - \varepsilon$. By letting $\alpha_1 = 1 - \varepsilon - 3\alpha/2$ and $\alpha_2 = \alpha/2$, we establish the lemma.

7.5.2. Removing zerocheck *Gates.* We move on to take care of zerocheck gates.

LEMMA 7.15 (REMOVING zerocheck GATES). *Let $p > (s + 1) \cdot 2^{s+1}$. For every $\beta > 0$, there exists a* $\mathrm{poly}(s, \log p, \log(1/\beta))$*-time computable probabilistic mapping f that maps an s-size arithmetic circuit $P : \mathbb{F}^n \to \mathbb{F}^m$ over* {zerocheck, add, subtract, multiply, divide, 0, 1} *into an $(s + s^2)$-size arithmetic circuit $f(P) : \mathbb{F}^{n+1} \to \mathbb{F}^{m+1}$ over* {add, subtract, multiply, divide, 0, 1} *such that the following hold for every s-size strictly arithmetic circuit T:*

—*If (P, T) is a Yes instance of $\mathsf{AP}_{d,\varepsilon,p,\Delta}$, then, with probability $1 - \beta$, the pair $(f(P), T)$ is a Yes instance of $\mathsf{AP}_{d',\varepsilon',p,\Delta}$;*

—*If* (P, T) *is a No instance of* $\mathsf{AP}_{d,\varepsilon,p,\Delta}$*, then, with probability* $1-\beta$*, the pair* $(f(P), T)$ *is a No instance of* $\mathsf{AP}_{d',\varepsilon',p,\Delta}$*,*

where $d' = d + 1$ *and* $\varepsilon' = \varepsilon + \frac{s2^{s+1}}{p}$.

Note that after the removal of zerocheck gates, the mapping P may contain a division by zero. As usual, in this case, the output may be arbitrary. We will show that our reduction works regardless of the behavior of the circuit in this case.

PROOF. We will use a zerocheck-removal procedure defined in Proposition 3.12. Recall that this procedure outputs, with probability $1-\beta$, a zerocheck-free circuit $\hat{P} : \mathbb{F}^n \to \mathbb{F}^m$ of size s together with a strictly arithmetic circuit $G : \mathbb{F}^n \to \mathbb{F}$ of size s^2 that computes a degree $s2^{s+1}$ polynomial such that P and $\hat{P}$ agree on all inputs x that are not roots of G. Based on $\hat{P}$ and G, we define a new zerocheck-free circuit with $n+1$ inputs and $m+1$ outputs. The first m outputs are computed by $\hat{P}(x_1, \ldots, x_n)$. The last $(m+1)$-th output is the polynomial $\hat{G}(x_1, \ldots, x_{n+1}) = G(x_1, \ldots, x_n) \cdot x_{n+1}$, where x_{n+1} is a new input variable. (Intuitively, when x_{n+1} is random, $\hat{G}$ contains a single bit of information that signals whether $\hat{P}$ equals to P.)

We analyze the effect of this procedure for Yes instances (assuming that the algorithm from Proposition 3.12 succeeds).

CLAIM 7.16. *If* (P, T) *is a Yes instance of* $\mathsf{AP}_{d,\varepsilon,p,\Delta}$*, then* $((\hat{P}, \hat{G}), T)$ *is a Yes instance of* $\mathsf{AP}_{d',\varepsilon',p,\Delta}$.

PROOF. Since (P, T) is a Yes instance, there exists a pair of degree d nonzero polynomials $A, B : \mathbb{F}^m \to \mathbb{F}$ such that $\frac{A(P(\boldsymbol{x}))}{B(P(\boldsymbol{x}))} = T(\boldsymbol{x})$ for every $\boldsymbol{x}$ for which $B(P(\boldsymbol{x})) \neq 0$.

Note that A and B are defined over m input variables $z = (z_1, \ldots, z_m)$. We extend these polynomials with an additional input variable z_{m+1} via

$$A'(z, z_{m+1}) = A(z) \cdot z_{m+1}, \quad \text{and} \quad B'(z, z_{m+1}) = B(z) \cdot z_{m+1}.$$

Observe that

$$\begin{aligned}
\frac{A'(\hat{P}(\boldsymbol{x}), \hat{G}(\boldsymbol{x}))}{B'(\hat{P}(\boldsymbol{x}), \hat{G}(\boldsymbol{x}))} &= \frac{A(\hat{P}(\boldsymbol{x})) \cdot \hat{G}(\boldsymbol{x})}{B(\hat{P}(\boldsymbol{x}) \cdot \hat{G}(\boldsymbol{x})} \\
&= \frac{A(\hat{P}(\boldsymbol{x}))}{B(\hat{P}(\boldsymbol{x}))} \\
&= \frac{A(P(\boldsymbol{x}))}{B(P(\boldsymbol{x}))} = T(\boldsymbol{x}),
\end{aligned}$$

where the second equality holds for every $\boldsymbol{x}$ that is not a root of $\hat{G}$, the third equality holds for every $\boldsymbol{x}$ that is not a root of G (since P and $\hat{P}$ agree on all these points), and the last equality holds for every $\boldsymbol{x}$ for which $B(P(\boldsymbol{x})) \neq 0$ (by our assumption). Overall, equality holds for every $\boldsymbol{x}$ that does not zero the denominator $B'(\hat{P}(\boldsymbol{x}), \hat{G}(\boldsymbol{x}))$, as required. Finally, A' and B' are of degree at most $d+1$ and so the claim follows. □

We move on to analyze No instances.

CLAIM 7.17. *If* (P, T) *is a No instance of* $\mathsf{AP}_{d,\varepsilon,p,\Delta}$*, then* $((\hat{P}, \hat{G}), T)$ *is a No instance of* $\mathsf{AP}_{d',\varepsilon',p,\Delta}$.

PROOF. Since (P, T) is a No instance, the distribution $(P, T)(U_n)$ is ε-close to some distribution $(\mathbf{P}', \mathbf{T}')$ with $\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') > \log p - \Delta$. Consider the distribution $((\mathbf{P}', \mathbf{G}'), \mathbf{T}')$

obtained from $(\mathbf{P}', \mathbf{T}')$ by concatenating an independent uniform element $\mathbf{G}' \xleftarrow{R} \mathbb{F}$. Since $\mathbf{G}'$ is independent of $\mathbf{T}'$, it still holds that $\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}', \mathbf{G}) > \log p - \Delta$ (see Fact 3.1 item 2). We will show that $((\hat{P}, \hat{G}), T)(U_n)$ is $(\varepsilon + \frac{s2^{s+1}}{p})$-close to $((\mathbf{P}', \mathbf{G}'), \mathbf{T}')$.

Consider the "hybrid" random variable $((P, x_{n+1}), T)(U_{n+1})$, which for $\boldsymbol{x} \xleftarrow{R} \mathbb{F}^{n+1}$ outputs the tuple $(P(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n), \boldsymbol{x}_{n+1}, T(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n))$. This random variable is distributed identically to $((\hat{P}, \hat{G}), T)(\boldsymbol{x})$, conditioned on the event $G(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \neq 0$. By Schwartz-Zippel, this event happens with probability at most $\frac{s2^{s+1}}{p}$, and therefore the two distributions are $\frac{s2^{s+1}}{p}$-close.

In addition, it is not hard to verify that the distributions $((P, x_{n+1}), T)(U_{n+1})$ and $((\mathbf{P}', \mathbf{G}'), \mathbf{T}')$ are ε-close. (Indeed, $((P, x_{n+1}), T)(U_{n+1})$ is obtained from $(P, T)(U_n)$ via the same randomized mapping that derives $((\mathbf{P}', \mathbf{G}'), \mathbf{T}')$ from $(\mathbf{P}', \mathbf{T}')$.) It follows, by the transitivity of statistical distance, that $((\hat{P}, \hat{G}), T)(U_n)$ is $\varepsilon + \frac{s2^{s+1}}{p}$-close to $((\mathbf{P}', \mathbf{G}'), \mathbf{T}')$, and the claim follows. □

This completes the proof of Lemma 7.15

7.5.3. Removing Division Gates.

LEMMA 7.18 (REMOVING divide GATES). *There exists an efficiently computable (deterministic) mapping f that maps an s-size arithmetic circuit $P : \mathbb{F}^n \to \mathbb{F}^m$ over* {add, subtract, multiply, divide, 0, 1} *into a $4s$-size arithmetic circuit $f(P) : \mathbb{F}^{n+m} \to \mathbb{F}^{2m}$ over* {add, subtract, multiply, 0, 1} *such that the following hold for every s-size strictly arithmetic circuit T:*

—*If (P, T) is a Yes instance of $\mathsf{AP}_{d,\varepsilon,p,\Delta}$, then the pair $(f(P), T)$ is a Yes instance of $\mathsf{AP}_{d',\varepsilon,p,\Delta}$;*
—*If (P, T) is a No instance of $\mathsf{AP}_{d,\varepsilon,p,\Delta}$, then the pair $(f(P), T)$ is a No instance of $\mathsf{AP}_{d',\varepsilon,p,\Delta}$,*

where $d' = 2d + m$.

PROOF. We assume that P_i, the ith output of P, is described by a circuit of the form A_i/B_i, where A_i and B_i are arithmetic circuits of size $s' \leq 4s$ over {add, subtract, multiply, 0, 1} that compute polynomials of degree at most 2^s. This is without loss of generality, since any arithmetic circuit P_i over {add, subtract, multiply, divide, 0, 1} can be transformed into this form at the expense of increasing its size by a factor of 4 (cf. Shpilka and Yehudayoff [2010, Proof of Thm. 2.11]). We define a new instance to the problem by keeping the same target polynomial T and modifying P to $f(P) = (A'_i, B'_i)_{i=1}^m$, where

$$A'_i(x_1, \ldots, x_n, r_i) = A_i(x) \cdot r_i, \quad B'_i(x_1, \ldots, x_n, r_i) = B_i(x) \cdot r_i,$$

and $r = (r_1, \ldots, r_m)$ are new input variables. (Formally, we redefine T as $T(x, r) = T(x)$.)

We claim that if the original instance was a Yes instance, then the new instance is also a Yes instance. Assume that there exists $Q = A/B$ such that $Q(P(\boldsymbol{x})) = T(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{F}^n$ that are not roots of B. Define the rational function $Q'(A'_1, B'_1, \ldots, A'_m, B'_m)$, which outputs $Q(A'_1/B'_1, \ldots, A'_m/B'_m) \cdot (\prod_i B'_i)/(\prod_i B'_i)$. For every $\boldsymbol{x}, \boldsymbol{r}$ that is not a root of $\prod_i B'_i$, we have that $Q'(f(P)(\boldsymbol{x}, \boldsymbol{r})) = Q(P(\boldsymbol{x}))$. Observe that $\boldsymbol{x}, \boldsymbol{r}$ that are roots of $\prod_i B'_i$ are, by construction, roots of the denominator of Q'. It follows that $(f(P), T)$ is indeed a Yes instance of $\mathsf{AP}_{d',\varepsilon',p,\Delta'}$, where the degree d' of the nominator and denominator of Q (viewed as polynomials in $2m$ variables) is at most $d' = 2d + m$.

We move on to analyze the case of a No instance. Assume that the distribution $(P, T)(U_n)$ is ε-close to some distribution $(\mathbf{P}', \mathbf{T}')$ with $\tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') > \log p - \Delta$. Consider the randomized mapping g, which, given a vector $p \in \mathbb{F}^m$, samples $r \xleftarrow{R} \mathbb{F}^m$ and outputs the vector $(p_1 r_1, r_1, \ldots, p_m r_m, r_m) \in \mathbb{F}^{2m}$. It is not hard to verify that the distribution $(g(P(U_n)), T(U_n))$ is distributed identically to $(f(P)(U_n), T(U_n))$. It follows that $(f(P)(U_n), T(U_n))$ is ε-close to $(g(\mathbf{P}'), \mathbf{T}')$. Finally, by Fact 3.1 (item 2), $\tilde{\mathbf{H}}_\infty(\mathbf{T}'|g(\mathbf{P}')) > \tilde{\mathbf{H}}_\infty(\mathbf{T}'|\mathbf{P}') > \log p - \Delta$. Hence, $(f(P), T)$ is a No instance of $\mathsf{AP}_{d',\varepsilon,p,\Delta}$. The lemma follows. □

7.5.4. Putting It All Together (Proving Theorem 7.2, Part 2). We employ Lemmas 7.13, 7.15, and 7.18 to reduce an instance (P, T) of $\mathsf{AP}_{d,\varepsilon,p,\Delta}$ to a strictly arithmetic instance of $\mathsf{AP}_{d',\varepsilon',\Delta',p}$ and then apply the first part of Theorem 7.2. Details follow.

Given an s-size arithmetic circuit $P : \mathbb{F}^n \to \mathbb{F}^m$ and an s-size strictly arithmetic circuit $T : \mathbb{F}^n \to \mathbb{F}$, we transform P into a strictly arithmetic circuit P' as follows: (1) change the sample gates to be input gates, (2) remove the bitsample gates by applying Lemma 7.13 with $\alpha = \frac{2}{3}(\frac{1}{5} - \varepsilon)$, (3) remove the zerocheck gates by applying Lemma 7.15 with $\beta = 2^{-s}$, and (4) remove the divide gates by applying Lemma 7.18. The resulting circuit P' has $n' = n + m + 2 \le 2s$ inputs, $m' = 2m + 2 \le 2s$ outputs, and size $s' = 4(s + s^2) \le 5s^2$. Let

$$d' = 2d + m + 3, \qquad \varepsilon' = \frac{2\varepsilon}{1 - \varepsilon - 3\alpha/2} + \frac{s2^{s+1}}{p}, \qquad \Delta' = \Delta + s + 1 + \log(1/\alpha).$$

If (P, T) is a Yes instance, then, with probability $1 - \beta$, the pair (P', T) is a Yes instance of $\mathsf{AP}_{d',\varepsilon',\Delta',p}$. If (P, T) is a No instance, then, with probability $\alpha - \beta$, the pair (P', T) is a Yes instance of $\mathsf{AP}_{d',\varepsilon',\Delta',p}$.

For $\alpha = \frac{2}{3}(\frac{1}{5} - \varepsilon)$ and $p \ge (d + s)^{2s} \cdot 2^{3s+\Delta}$, it holds that $\varepsilon' < \frac{1}{2}$ and $p > \frac{(n'+1)d^{n'+1}(1+2^{\Delta'})}{(1/2-\varepsilon')^2}$ (for all sufficiently large s). Therefore, the first part of Theorem 7.2 guarantees an algorithm $\mathcal{A}$ that decides whether (P', T) is Yes instance or a No instance of $\mathsf{AP}_{d',\varepsilon',\Delta',p}$ with an error of β in time $\mathrm{poly}(s', \log(1/\beta), \log(p))$. (The low error can be obtained via standard amplification techniques.) We output the value $\mathcal{A}(P', T)$.

Overall, if (P, T) is a Yes instance, then we answer correctly with probability $1 - 2\beta$, and if (P, T) is a No instance, then we answer correctly with probability $\alpha' = \alpha - 2\beta$. We repeat the previous procedure $t = \ln(1/\beta)/\alpha' = \mathrm{poly}(s)$ times (with independent randomness) and output "No" if at least one of the iterations outputs "No." We will err on a No instance with probability $(1 - \alpha')^t < \beta = 2^{-s}$ and err on a Yes instance with probability $3t\beta = 2^{-\Omega(s)}$. This completes the proof of the second part of Theorem 7.2. □

POSITIVE RESULTS

In the following sections, we give constructions of cryptographic primitives in the arithmetic model. We start by introducing the RLC assumption on which we base our constructions (Section 8) and then provide constructions for an arithmetic pseudorandom generator (Section 9), symmetric and public key encryption (Section 10), commitments (Section 11), and protocols for secure computation (Section 12).

As explained in the introduction, we present several alternative constructions based on three different approaches: (1) an abstract primitive (Arithmetic/Binary one-time encryption), which allows importing binary constructions to the arithmetic setting; (2) direct approach, which adopts known LPN-based construction to the arithmetic setting; and (3) a reduction-based approach, which exploits the fact that some classical cryptographic transformations have a simple arithmetic variant.

Remark 7.19 (*Adversarial Model*). From now on, we move to a nonuniform model and use the term *efficient adversary* to denote a probabilistic polynomial-time (binary) circuit family. This choice is taken to simplify the presentation, and all our results can be easily adopted to the uniform model.

8. THE RLC ASSUMPTION

We present the RLC assumption of Ishai et al. [2009], which asserts that a noisy codeword of a random linear code is pseudorandom.

Notation. For a field $\mathbb{F}$, integer ℓ, and real number $p \in (0, 1)$, let $\chi_p^{\ell}(\mathbb{F})$ denote the probability distribution over $\mathbb{F}^{\ell}$ where each coordinate takes the value zero with probability $1-p$ and a random element in $\mathbb{F}$ with probability p. When the field is clear from the context, we omit it and write χ_p^{ℓ}. We let $\mathcal{D}_{\mathbb{F},p}^{\ell\times n}$ denote the probability distribution over pairs $(M, v) \in \mathbb{F}^{\ell(n)\times n} \times \mathbb{F}^{\ell(n)}$ in which $M \xleftarrow{R} \mathbb{F}^{\ell(n)\times n}$ and $c = M \cdot s + e$, where $s \xleftarrow{R} \mathbb{F}^n$ and $e \xleftarrow{R} \chi_p^{\ell}$. When the parameters $\ell = \ell(n)$, $p = p(n)$ and $\mathbb{F} = \{\mathbb{F}_n\}$ are inddexed by n, we let $\mathcal{D}_{\mathbb{F},p}^{\ell\times n}$ denote the corresponding distribution ensemble.

ASSUMPTION 8.1 (RLC(n, ℓ, p)). *For security parameter n, length parameter $\ell(n)$, and noise parameter $p(n) \in (0, 1)$, the* RLC(n, ℓ, p) *assumption asserts that for every efficient adversary $\mathcal{A}$ the probability of winning the following game is at most* $\frac{1}{2} + \mathrm{neg}(n)$*:*

IND-Game(1^n):

—*$\mathcal{A}$ receives 1^n, chooses a field's implementation $\mathbb{F}$, and sends $\mathbb{F}$ to the challenger.*
—*The challenger samples a challenge bit $b \xleftarrow{R} \{0, 1\}$. If $b = 0$, the challenger sends to $\mathcal{A}$ a uniformly chosen matrix-vector pair $(M, c) \xleftarrow{R} (\mathbb{F}^{\ell\times n}, \mathbb{F}^{\ell})$. If $b = 1$, the challenger sends the pair $(M, c) \xleftarrow{R} \mathcal{D}_{\mathbb{F},p}^{\ell\times n}$, where $\ell = \ell(n)$ and $p = p(n)$.*
—*$\mathcal{A}$ outputs b' and wins if $b = b'$.*

Remarks:

(1) Observe that the RLC(n, ℓ, p) assumption is equivalent to the assumption that for every efficient field family $\mathbb{F} = \{\mathbb{F}_n\}$, it holds that $\mathcal{D}_{\mathbb{F},p}^{\ell\times n} \stackrel{c}{\approx} (M \xleftarrow{R} \mathbb{F}_n^{\ell\times n}, v \xleftarrow{R} \mathbb{F}_n^{\ell})$.
(2) The error distribution χ_p^{ℓ} can be efficiently sampled up to a negligible statistical distance by an arithmetic circuit that receives as input $(H_2(p) + \varepsilon) \cdot \ell$ random bits and $(p + \varepsilon) \cdot \ell$ random field elements. The circuit first chooses the subset of noisy coordinates $b = (b_1, \ldots, b_\ell) \in \{0, 1\}^{\ell}$ by sampling ℓ independent Bernoulli random variables with mean p. (This step can be implemented by a binary circuit that uses $H_2(p)+\varepsilon$ random input bits, and so it can be emulated by an arithmetic circuit that takes as input only random bits.) Once the noisy coordinates are selected, a random noise from $\mathbb{F}$ is assigned to each of them; with overwhelming probability there will be less than $(p + \varepsilon) \cdot \ell$ noisy coordinates and so this step can be implemented using only $(p + \varepsilon) \cdot \ell$ random field elements.
(3) It is not hard to see that RLC can only become harder when the noise p increases and the length ℓ decreases (see Proposition 8.2). Hence, it is desirable to use the assumption with high noise and short matrices.

We put forward the following simple proposition for future reference.

PROPOSITION 8.2. *Assume that the* RLC(n, ℓ, p) *assumption holds for polynomially bounded $\ell(n)$ and efficiently computable noise rate $p(n) < 1/2$. Then,*

(*1*) RLC(n, ℓ, P) *holds for any efficiently computable $P(n) \geq p(n)$.*
(*2*) RLC(n, L, p) *holds for any $L(n) \leq \ell(n)$.*
(*3*) RLC$(N(n), \ell(n), p(n))$ *holds for any $n \leq N(n) \leq n^c$, where c is an arbitrary constant.*

PROOF. (1) Observe that a vector from $\chi_{p'}^{\ell}$ can be sampled by adding a vector from χ_p^{ℓ} to a vector from χ_α^{ℓ}, where $\alpha = (p' - p)/(1 - 2p)$. Consider the mapping T_1, which maps $(M, c) \in (\mathbb{F}^{\ell \times n} \times \mathbb{F}^{\ell})$ to the pair $(M, c + e')$, where $e' \in \mathbb{F}^{\ell}$ is sampled from χ_α^{ℓ}. Note that T_1 takes the distribution $\mathcal{D}_{\mathbb{F},p}^{\ell \times n}$ to $\mathcal{D}_{\mathbb{F},p'}^{\ell \times n}$ and takes the uniform distribution (over $\mathbb{F}^{\ell \times n} \times \mathbb{F}^{\ell}$) to itself. Since T_1 is computable in polynomial time (in n), a distinguisher for $\mathsf{RLC}(n, \ell, P)$ implies a distinguisher for $\mathsf{RLC}(n, \ell, p)$.

(2) Consider the mapping T_2 that, given an input $(M, c) \in (\mathbb{F}^{\ell \times n} \times \mathbb{F}^{\ell})$, outputs the matrix M', which consists of the first L rows of M, and the vector c', which consists of the first L entries of c. Then T_2 takes the distribution $\mathcal{D}_{\mathbb{F},p}^{\ell \times n}$ to $\mathcal{D}_{\mathbb{F},p}^{L \times n}$ and takes the uniform distribution over $\mathbb{F}^{\ell \times n} \times \mathbb{F}^{\ell}$ to the uniform distribution over $\mathbb{F}^{L \times n} \times \mathbb{F}^{L}$. Since T_2 is computable in polynomial time (in n), a distinguisher for $\mathsf{RLC}(n, L, p)$ implies a distinguisher for $\mathsf{RLC}(n, \ell, p)$.

(3) Consider the mapping T_3, which, given an input $(M, c) \in (\mathbb{F}^{\ell \times n} \times \mathbb{F}^{\ell})$, samples $s' \xleftarrow{R} \mathbb{F}^{N-n}$ and $M' \xleftarrow{R} \mathbb{F}^{\ell \times (N-n)}$, and returns $((M|M'), c + M's')$. It can be seen that T_3 takes the distribution $\mathcal{D}_{\mathbb{F},p}^{\ell \times n}$ to $\mathcal{D}_{\mathbb{F},p}^{\ell \times N}$ and takes the uniform distribution over $\mathbb{F}^{\ell \times n} \times \mathbb{F}^{\ell}$ to the uniform distribution over $\mathbb{F}^{\ell \times N} \times \mathbb{F}^{N}$. Therefore, a poly($N$)-time distinguisher for $\mathsf{RLC}(N, \ell, p)$ implies a poly(N)-time distinguisher for $\mathsf{RLC}(n, \ell, p)$. Since N is polynomial in n, the proposition follows. $\square$

9. ARITHMETIC PSEUDORANDOM GENERATOR

In this section, we construct an APRG with arbitrary polynomial stretch based on the RLC assumption.

9.1. Basic Observations

Recall that $\mathsf{APRG} = \{\mathsf{APRG}_n\}$ is viewed as a randomized circuit that samples a pseudorandom distribution over $\mathbb{F}^{\ell(n)}$ using n randomized gates (see Definition 4.1). Following our general convention, we allow the PRG to use both sample gates and bitsample gates. (This is also motivated by our applications that can tolerate the use of such gates.) In the following, we observe that the number of bitsample gates can be always reduced via the use of a binary PRG.

OBSERVATION 9.1. *Assume that* APRG $= \{\mathsf{APRG}_n\}$ *outputs $\ell(n)$ pseudorandom field elements (as per the second item of the previous definition) and uses $n < \ell(n)$ gates that sample random field elements (*sample*) and $k(n)$ gates that sample random bits (*bitsample*), where $k(n)$ may be larger than $\ell(n)$. Then, for every constant $\varepsilon > 0$, there exists an arithmetic PRG* $\mathsf{APRG}' = \{\mathsf{APRG}'_n\}$ *that samples $\ell(n)$ pseudorandom field elements and uses n* sample *gates and n^{ε}* bitsample *gates. Furthermore, if* APRG *is simple (i.e., does not contain division or* zerocheck *gates), then so is* APRG$'$.

PROOF. We use a binary PRG that stretches n^{ε} bits to $k(n)$ pseudorandom bits. The new APRG samples n^{ε} bits, feeds them to the PRG, and uses the $k(n)$ outputs instead of the original bitsample gates. It is not hard to show that the outcome is pseudorandom (otherwise, one can break the PRG). Furthermore, the PRG can be written in arithmetic form by arithmetizing the Boolean logic (i.e., replace AND with multiplication and NOT(w) with $1 - w$). Finally, the existence of a binary PRG follows from the hypothesis of the theorem. Indeed, a binary PRG can be derived directly by instantiating an APRG with a sufficiently large field $|\mathbb{F}| > 2^{2\ell}$ and by using a proper implementation of the field (i.e., that guarantees that a random field element is represented by a uniform, or close to uniform, string). $\square$

We do not know whether it is possible to completely eliminate bitsample gates. The existence of an arithmetic PRG that does not use random bits at all is left as an interesting open problem.

Length expansion. We continue by showing that given a minimal APRG that expands n inputs to $n+1$ outputs, one can construct an APRG$'$ with polynomial expansion n^c. The transformation is similar to the standard transformation from the binary setting [Goldreich 2001, Chapter 3.3].

LEMMA 9.2 (APRG EXPANSION). *Suppose that there exists an Arithmetic PRG* APRG *with an additive expansion of* 1*. Then, for any polynomial* $q(n) = \mathrm{poly}(n)$*, there exists an arithmetic PRG* APRG$'$ *with an output length of* $q(n)$*, where* n *denotes the seed length. Furthermore, if* APRG *is simple, then so is* APRG$'$.

PROOF SKETCH. In the following, we assume, WLOG, that the underlying APRG G uses n_1 sample gates and n_2 bitsample gates, where $n_1 + n_2 = n$. Furthermore, we view the random gates of the APRG as input gates, and accordingly view the APRG as a function $G : \mathbb{F}^{n_1} \times \{0,1\}^{n_2} \to \mathbb{F}^{n+1}$. We will construct a new function APRG$'$: $\mathbb{F}^{n_1} \times \{0,1\}^{q(n)\cdot n_2} \to \mathbb{F}^{q(n)}$ whose output is pseudorandom when evaluated on a random input. Then, we use Observation 9.1 to reduce the number of binary inputs to n_2. The function APRG$'$ is defined iteratively.

—Given a seed $a_0 \in \mathbb{F}^{n_1}$ and $b = (b_0, \ldots, b_{q-1})$ where $b_i \in \{0,1\}^{n_2}$.
—For $i = 0$ to $q-1$: Compute $G(a_i, b_i) \in \mathbb{F}^{n+1}$ and parse the result as $(a_{i+1}, y_{i+1}) \in \mathbb{F}^{n_1} \times \mathbb{F}^{(n+1)-n_1}$.
—Output $y_1, \ldots, y_q$.

The proof of security follows from a standard hybrid argument. For $j = 0, \ldots, q$, define the hybrid H_j in which for every $i \le j$ the values $(a_i, y_i) \xleftarrow{R} \mathbb{F}^{n_1} \times \mathbb{F}^{(n+1)-n_1}$ are uniformly chosen, and the other iterations remain unchanged. The hybrid H_0 corresponds to the real construction, while the outcome of the last hybrid H_q is clearly uniform. We show that H_0 is indistinguishable from H_q by showing that a distinguisher $\mathcal{A}$ that ε-distinguishes these distributions can be used to violate the pseudorandomness of G. Given a challenge $z \in \mathbb{F}^{n+1}$, we sample a random location $j \in \{0, \ldots, q-1\}$, sample the first $j-1$ values $(a_i, y_i)_{i<j}$ uniformly, let $(a_j, b_j) = z$, and continue for the other iterations as in the real constructions. The output is given to the distinguisher $\mathcal{A}$. It is not hard to show that the distinguishing advantage of the new distinguisher is ε/q, and so the lemma follows. □

9.2. Construction Based on RLC

We continue with an arithmetic construction of an APRG based on the RLC assumption. Recall that the RLC(n, ℓ, p) assumption asserts that for a random matrix $M \xleftarrow{R} \mathbb{F}^{\ell \times n}$, a random vector $s \xleftarrow{R} \mathbb{F}^n$, and an error vector $e \xleftarrow{R} \chi_p^\ell(\mathbb{F})$, the output $(M, Ms + e)$ is pseudorandom (see Assumption 8.1). This gives an immediate construction of an APRG.

THEOREM 9.3. *Assuming* RLC$(n, \ell = \frac{n}{1-p-\varepsilon}, p)$ *holds for some constants* $p \in (0,1), \varepsilon > 0$*, there exists a simple APRG with an arbitrary polynomial output length.*

PROOF. The circuit samples a random matrix $M \xleftarrow{R} \mathbb{F}^{\ell \times n}$ and a random vector $s \xleftarrow{R} \mathbb{F}^n$ and computes $y = Ms$. The noise is sampled as described in Section 8. First, we use a polynomial number of bitsample gates to sample a binary vector of ℓ independent Bernoulli random variables $\alpha = (\alpha_1, \ldots, \alpha_\ell)$, each with a probability of success p. If the number of 1s, t, is smaller than $(p+\varepsilon)\ell$, we sample t random field elements and place them in the 1s' locations of α; the resulting noise vector e is then added to y and the output is $(M, Ms+e)$. If $t \geq (p+\varepsilon)\ell$, we let e be the all-zero vector. By Chernoff bound, the latter event happens with probability $2^{-\Omega(\ell)}$, and so the resulting distribution is statistically close to the RLC distribution, which is pseudorandom by assumption. It follows that the output of the generator is also pseudorandom. Since the number of random field elements in the seed is strictly smaller than $\ell n + n + (p+\varepsilon)\ell$, which is upper bounded by the output length $\ell n + \ell$, we can apply Observation 9.1 and Lemma 9.2 and get an APRG with an arbitrary polynomial output length. $\square$

10. ENCRYPTION

In this section, we construct arithmetic encryption schemes in the public-key and symmetric-key setting based on an RLC assumption. We begin with a construction of a one-time secure symmetric encryption scheme, which encrypts field elements using binary keys. We use this *arithmetic/binary encryption* scheme to obtain CPA-secure arithmetic encryption schemes in the symmetric-key and public-key settings (Section 10.2). Finally, we show that known LPN-based constructions arithmetize and yield arithmetic symmetric encryption schemes (Section 10.3) and public-key encryption schemes (Section 10.4).

10.1. One-Time Secure Arithmetic/Binary Encryption

Recall that *arithmetic/binary encryption* is an arithmetic symmetric encryption scheme with one-time security with the additional property that all the elements of the secret key sk are taken from the subset $\{0, 1\} \subset \mathbb{F}$. That is, the key is essentially a string of bits given as a sequence of 0-1 field elements. (See Definition 4.2.) As we will see, this special property can be used as a bridge from the binary model to the arithmetic model. We begin with a simple construction of ABE based on the RLC assumption.

Notation. For an integer ℓ and a real number $p \in (0, 1)$, let $\mathcal{B}_p^\ell$ denote the probability distribution over $\{0, 1\}^\ell$ where each coordinate takes the value 1 with probability p.

CONSTRUCTION 10.1 (ARITHMETIC/BINARY ENCRYPTION). *We parameterize our construction by a constant $p \in (0, 1)$ and a constant $\varepsilon > 0$. For a security parameter n, we set the length parameter $\ell := \frac{(1+\varepsilon)n}{1-p}$. The scheme consists of the following circuits:*

—KGen_n: *Sample a binary*[20] *noise vector $e \xleftarrow{R} \mathcal{B}_p^\ell$.*

—$\mathsf{Enc}_n^{\mathbb{F}}(e, x)$: *Sample $A \xleftarrow{R} \mathbb{F}^{\ell\times n}$, $s \xleftarrow{R} \mathbb{F}^n$, and $u \xleftarrow{R} \mathbb{F}^\ell$. Output $(A, c := As + u \otimes e + x^\ell)$, where x^ℓ denotes the ℓ long (column) vector $(x, \ldots, x)$ and $\otimes$ stands for entry-wise product.*

—$\mathsf{Dec}_n^{\mathbb{F}}(e, (A, c))$: *Find an assignment to the variables $y, z = (z_1, \ldots, z_n)$ that satisfies the linear system $A'z + y^{\ell'} = c'$, where A' and c' are the restrictions of A and c to the ℓ' nonnoisy coordinates of e, and output the value that is assigned to y.*

We notice that the decryption can be implemented arithmetically by solving the system of linear equations $(I - E)c = (I - E)(Az + y^\ell)$, where E is the diagonal matrix with $E_{ii} = e_i$.

The following two lemmas establish the correctness and the security of the scheme.

[20]The reader should note that in this subsection, e denotes a *binary* vector and not a vector of field elements (as in other sections).

LEMMA 10.2 (CORRECTNESS). *For every field $\mathbb{F}$, and for all $x \in \mathbb{F}$, it holds that*

$$\Pr_{e \xleftarrow{R} \mathsf{KGen}_n} \left[\mathsf{Dec}_n^{\mathbb{F}}\left(e, \mathsf{Enc}_n^{\mathbb{F}}(e, x)\right) = x\right] \geq 1 - \mathrm{neg}(n).$$

PROOF. Consider the set of linear equations solved at decryption

$$c' = A'z + y^{\ell'}. \tag{21}$$

Clearly $z = s$ and $y = x$ is a valid solution for Equation (21). Next, we claim that (1) if there exists a vector $v \in \mathbb{F}^{\ell'}$ in the left nullspace of A' whose entries do not sum to zero ($\sum_i v_i \neq 0$), then any solution (z, y) to Equation (21) satisfies $y = x$ and so decryption succeeds, and (2) such a vector v exists with all but exponentially small probability.

Indeed, to see (1), observe that any solution to Equation (21) also satisfies the equation

$$vc' = v(A'z + y^{\ell'}) = y \sum_{i=1}^{\ell'} v_i,$$

and so y is determined uniquely via $vc' / \sum_{i=1}^{\ell'} v_i$. To prove (2), observe that if v of the required form does not exist, then the all-one vector $1^{\ell'}$ is spanned by the columns of A'. Let us condition on the event that ℓ', the number of nonnoisy coordinates in e, is at least $(1 + \varepsilon/2)n$. By a Chernoff bound, this event happens with all but exponentially small probability. Recalling that A' is uniformly distributed over $\mathbb{F}^{\ell' \times n}$, we conclude that the probability that A' spans the all-one vector is at most $|\mathbb{F}|^{n-\ell'}$, which is negligible when $\ell' > (1 + \varepsilon/2)n$. The lemma follows. □

LEMMA 10.3 (SECURITY). *Let $p \in (0, 1)$ and $\varepsilon > 0$ be some constants. Suppose that the* $\mathsf{RLC}(n, \ell = \frac{(1+\varepsilon)n}{1-p}, p)$ *assumption holds; then Construction 10.1 (instantiated with the same p and ε) is one-time computationally secure.*

PROOF. Assume toward a contradiction that an efficient adversary $\mathcal{A}$ wins the *One-Time IND game* with probability $1/2 + \delta(n)$ for some nonnegligible function δ. We define a new adversary $\mathcal{A}'$ against the RLC assumption in the following way:

Adversary $\mathcal{A}'$:

—Initialize $\mathcal{A}$ and get a field $\mathbb{F}$ from it. Send the same $\mathbb{F}$ to the challenger.
—Receive from the challenger $(A, v) \in (\mathbb{F}^{\ell \times n}, \mathbb{F}^{\ell})$.
—Receive from $\mathcal{A}$ a pair $x_0, x_1 \in \mathbb{F}$.
—Sample $b \in \{0, 1\}$, and send to $\mathcal{A}$ the challenge $C = (A, v + x_i^{\ell})$.
—If $\mathcal{A}$ wins, return 0; otherwise, return 1.

If $\mathcal{A}'$ receives $(A, v) \xleftarrow{R} \mathcal{D}_{\mathbb{F},p}^{\ell \times n}$, then C is distributed as a random cyphertext and so by assumption $\mathcal{A}$ wins with the probability $\frac{1}{2} + \delta(n)$, and $\mathcal{A}'$ outputs 0. On the other hand, if $(A, v) \xleftarrow{R} (\mathbb{F}^{\ell \times n}, \mathbb{F}^{\ell})$, then by the security of the one-time pad, $\mathcal{A}$ outputs 0 with probability exactly 1/2. Overall, $\mathcal{A}'$ distinguishes $\mathcal{D}_{\mathbb{F},p}^{\ell \times n}$ from $(\mathbb{F}^{\ell \times n}, \mathbb{F}^{\ell})$ with the nonnegligible advantage $\frac{\delta(n)}{2}$. □

COROLLARY 10.4. *If, for some constants $p \in (0, 1)$, $\varepsilon > 0$, the assumption* $\mathsf{RLC}(n, \ell = \frac{(1+\varepsilon)n}{1-p}, p)$ *holds, then there exists an ABE scheme.*

Remark 10.5 (*Nontriviality*). The Learning Parity with Noise assumption is trivially hard (in an information-theoretic sense) when the noise rate is $\frac{1}{2}$. In contrast, the

$\mathsf{RLC}(n, \ell, p)$ assumption is nontrivial (i.e., implies the existence of a one-way function) for *any* constant value of $p \in (0, 1)$ as it should hold over fields of size larger than $1/p$. Indeed, when $\mathbb{F}$ is sufficiently large, the entropy needed to sample an $\mathsf{RLC}(n, \ell, p)$ instance (roughly $\log|\mathbb{F}| \cdot (\ell n + n + p\ell) + \ell \cdot H_2(p)$) is smaller than the entropy of the output ($\ell \log|\mathbb{F}|$) and so the assumption immediately implies the existence of a pseudorandom generator.

Remark 10.6 (*Weak Homomorphism*). Interestingly, Construction 10.1 is (weakly) homomorphic under addition and multiplication. Given a ciphertext $(A, c) \stackrel{R}{\leftarrow} \mathsf{Enc}_n^{\mathbb{F}}(e, x)$, one can create, for every message $y \in \mathbb{F}$ and every scalar $a \in \mathbb{F}$, a ciphertext $(A, c') \stackrel{R}{\leftarrow} \mathsf{Enc}_n^{\mathbb{F}}(e, a \cdot x + y)$ by computing $c' = a \cdot c + y^{\ell}$. This homomorphism is weak in the sense that the joint distribution $(A, c), (A, c')$ is statistically far from a fresh pair of ciphertexts $(\mathsf{Enc}_n^{\mathbb{F}}(e, x), \mathsf{Enc}_n^{\mathbb{F}}(e, a \cdot x + y)$. For this reason, our impossibility results (Section 7) do not apply here.

10.2. From ABE to Symmetric and Public-Key Encryption

Next, we show how to employ ABE (which offers only one-time security) in order to obtain a CPA-secure arithmetic encryption scheme either in the public or in the symmetric setting. The idea is to combine ABE and standard binary encryption via a hybrid mode.

CONSTRUCTION 10.7. *Let* $(\mathsf{BGen}, \mathsf{BEnc}, \mathsf{BDec})$ *be any binary CPA-secure encryption scheme (either in the symmetric setting or in the public-key setting) and let* $(\mathsf{ABGen}, \mathsf{ABEnc}, \mathsf{ABDec})$ *be any ABE. Consider the following arithmetic encryption scheme:*

—KGen_n: *Output* $(\mathsf{sk}, \mathsf{pk}) \stackrel{R}{\leftarrow} \mathsf{BGen}_n$.
—$\mathsf{Enc}_n^{\mathbb{F}}(\mathsf{pk}, x)$: *Sample* $e \stackrel{R}{\leftarrow} \mathsf{ABGen}_n$ *and output* $(c_1 \stackrel{R}{\leftarrow} \mathsf{BEnc}_n(\mathsf{pk}, e), c_2 \stackrel{R}{\leftarrow} \mathsf{ABEnc}_n^{\mathbb{F}}(e, x))$.
—$\mathsf{Dec}_n^{\mathbb{F}}(\mathsf{sk}, c_1, c_2)$: *Compute* $e' = \mathsf{BDec}_n(\mathsf{sk}, c_1)$ *and output* $x' = \mathsf{ABDec}_n^{\mathbb{F}}(e', c_2)$.

Correctness follows directly from the correctness of the underlying binary encryption scheme and ABE scheme.

LEMMA 10.8 (SECURITY). *Suppose that* $(\mathsf{BGen}, \mathsf{BEnc}, \mathsf{BDec})$ *is a CPA-secure public-key encryption scheme (symmetric encryption scheme, respectively) and* $(\mathsf{ABGen}, \mathsf{ABEnc}, \mathsf{ABDec})$ *is a one-time secure ABE; then Construction 10.7 is a CPA-secure public-key (symmetric key, respectively) arithmetic encryption scheme.*

PROOF. Let $\mathcal{A}$ be an adversary that attempts to win the standard IND-CPA game. To analyze $\mathcal{A}$'s success probability, we define (in Table II) several variants of the original game.

Let $\Pr[\langle \mathcal{A}, G_i \rangle = 1]$ be the probability that $\mathcal{A}$ returns 1 when interacting with the ith game; then the following holds:

(1) $|\Pr[\langle \mathcal{A}, G_1 \rangle = 1] - \Pr[\langle \mathcal{A}, G_2 \rangle = 1]| = \mathrm{neg}(n)$, or else one could break the CPA security of the binary encryption scheme on the challenge $(e, 0^{\ell})$.
(2) $|\Pr[\langle \mathcal{A}, G_2 \rangle = 1] - \Pr[\langle \mathcal{A}, G_3 \rangle = 1]| = \mathrm{neg}(n)$, or else one could break the one-time security of the arithmetic/binary encryption scheme on the challenge (x_0, x_1).
(3) $|\Pr[\langle \mathcal{A}, G_3 \rangle = 1] - \Pr[\langle \mathcal{A}, G_4 \rangle = 1]| = \mathrm{neg}(n)$, or else one could break the CPA security of the binary encryption scheme on the challenge $(0^{\ell}, e)$.

It follows, by a hybrid argument, that $|\Pr[\langle \mathcal{A}, G_1 \rangle = 1] - \Pr[\langle \mathcal{A}, G_4 \rangle = 1]| = \mathrm{neg}(n)$, and so $\mathcal{A}$'s wining probability in the original IND-CPA game is at most $1/2 + \mathrm{neg}(n)$. □

Table II. Hybrid Games

Game 1	Game 2	Game 3	Game 4
$(\mathsf{sk}, \mathsf{pk}) \xleftarrow{R} \mathsf{BGen}_n$ Query stage Challenge stage for (x_0, x_1): $e \xleftarrow{R} \mathsf{ABGen}_n$ $c_1 \xleftarrow{R} \mathsf{BEnc}_n(\mathsf{pk}, e)$ $c_2 \xleftarrow{R} \mathsf{ABEnc}_n^{\mathbb{F}}(e, x_0)$ Send $\mathcal{A}$ the challenge (c_1, c_2)	$c_1 \xleftarrow{R} \mathsf{BEnc}_n(\mathsf{pk}, 0^\ell)$	$c_2 \xleftarrow{R} \mathsf{ABEnc}_n^{\mathbb{F}}(e, x_1)$	$c_1 \xleftarrow{R} \mathsf{BEnc}_n(\mathsf{pk}, e)$

In each game, only the changes from the previous game are written. In the public-key setting, the key pk is given to the adversary at the beginning of the query stage, and in the private-key setting, we let $\mathsf{pk} = \mathsf{sk}$ and keep it hidden from the adversary. Recall that $\mathsf{sk}, \mathsf{pk}, e,$ and c_1 are binary vectors; c_2 is a vector of field elements; and x_0 and x_1 are field scalars.

Remark 10.9 (*Encrypting Long Messages*). Construction 10.7 allows one to encrypt a single field element. However, since we proved that the scheme is CPA secure, it can be extended to encrypt a sequence of field elements via standard concatenation as explained in Remark 4.3.

By combining the previous with Corollary 10.4, we derive a construction of a CPA-secure public-key arithmetic encryption scheme based on the RLC assumption and on a standard binary public-key encryption scheme. A similar result holds in the symmetric setting, except that in this case the existence of a binary CPA-secure encryption scheme already follows from the existence of (standard) one-way functions [Goldreich et al. 1986; Håstad et al. 1999], which in turn follows from the RLC assumption (see Remark 10.5). Overall, we obtain the following corollary.

COROLLARY 10.10. *Assuming* $\mathsf{RLC}(n, \ell = \frac{(1+\varepsilon)n}{1-p}, p)$ *holds for some constant* $p \in (0, 1)$ *and constant* $\varepsilon > 0$*, there exists a CPA-secure arithmetic symmetric encryption. Furthermore, if we additionally assume the existence of a standard public-key encryption scheme, then there exists a CPA-secure arithmetic public-key encryption scheme.*

10.3. Direct Construction of Symmetric Encryption

We proceed with a direct construction of symmetric encryption that generalizes the LPN-based construction of Gilbert et al. [2008] (see also Applebaum et al. [2009]). Although the underlying assumption is worse than the one used in Corollary 10.10, the scheme demonstrates our claim that LPN-based encryption arithmetizes.

CONSTRUCTION 10.11 (CPA-SECURE SYMMETRIC ENCRYPTION SCHEME). *We parameterize our construction by the constants* $p \in (0, \frac{1}{2})$ *and* $\varepsilon \in (0, \frac{1}{2} - p)$*. For a security parameter* n*, we set the length parameter* $\ell := \lceil \frac{np}{\varepsilon^2} \rceil$*. The scheme consists of the following circuits:*

—KGen_n*: Sample* $s \xleftarrow{R} \mathbb{F}^n$*, and let* $\mathsf{sk} = s$*.*
—$\mathsf{Enc}(s, x) : M \xleftarrow{R} \mathbb{F}^{n\times n}$, $e \xleftarrow{R} \chi_p^n(\mathbb{F})$*: Output* $(M, c := Ms + e + x^n)$*, where* x^n *denotes the* n*-long column vector* $(x, \ldots, x)$*.*
—$\mathsf{Dec}(s, (M, c))$*: Output the majority among the entries of the vector* $c - Ms$*.*

Note that the majority operation can be computed arithmetically via the use of zerocheck gates and by describing logical majority via arithmetic gates (e.g., by going through the standard AND, NOT Boolean basis and then replacing it with multiplication and subtraction). Also, as explained in Section 8, there exists an arithmetic circuit that samples from χ_p^ℓ, and so the construction is realizable in the arithmetic model.

Remark. As we require correctness to hold for all fields, we need to choose $p + \varepsilon < 1/2$, as otherwise, correctness would not hold over GF(2). Note that in settings where we are only interested in correctness for large fields, we can replace the restriction $p < 1/2$ with $p < 1$ and use a Reed-Solomon (RS) error-correcting code instead of the repetition code. (RS codes can be encoded and decoded arithmetically.) This modification allows us to base the construction on a seemingly more secure assumption (with larger noise level). One can also improve the rate and efficiency of the scheme by using RS codes and by standard amortization techniques similar to the ones described in Applebaum et al. [2009].

LEMMA 10.12 (SECURITY). *Suppose that the* $\mathsf{RLC}(n, \ell, p)$ *assumption holds for every polynomial* $\ell(n)$ *and for some constant* $p \in (0, 1/2)$*. Then Construction 10.11 (instantiated with the same* p*) is computationally secure.*

PROOF. Assume toward a contradiction that an efficient adversary $\mathcal{A}$ wins the IND-CPA game with probability $1/2 + \delta(n)$ for some nonnegligible function δ. Let $t = \text{poly}(n)$ be an upper bound on the running time of $\mathcal{A}$. We define a new adversary $\mathcal{B}$ against the RLC assumption in the following way:

Adversary $\mathcal{B}$:

—Initiate $\mathcal{A}$ and get $\mathbb{F}$ from it. Send the same $\mathbb{F}$ to the challenger.
—Receive from the challenger $(M, v) \in (\mathbb{F}^{t \cdot n \times n}, \mathbb{F}^{t \cdot n})$.
—Parse $M = (M_1, \ldots, M_t)$ and $v = (v_1, \ldots, v_t)$, where $M_i \in \mathbb{F}^{n \times n}$ and $v_i \in \mathbb{F}^n$.
—For the ith chosen plaintext query x of $\mathcal{A}$:
  —Send to $\mathcal{A}$ the pair $(M_i, v_i + x^n)$.
—For the challenge x_0, x_1:
  —Sample $b \in \{0, 1\}$.
  —Send to $\mathcal{A}$ the challenge $(M_t, v_t + x_b^n)$.
—If $\mathcal{A}$ wins, return 0; otherwise, return 1.

If the input (M, v) is sampled from the RLC distribution $\mathcal{D}_{\mathbb{F},p}^{tn \times n}$, then, by assumption, $\mathcal{A}$ wins with the probability $\frac{1}{2} + \delta(n)$, and $\mathcal{B}$ outputs 0. However, if $(M, v) \xleftarrow{R} (\mathbb{F}^{tn \times n}, \mathbb{F}^{tn})$, then the message is information-theoretically hidden from the adversary, and so $\mathcal{A}$ outputs 0 with probability exactly 1/2. Therefore, $\mathcal{B}$ distinguishes $\mathcal{D}_{\mathbb{F},p}^{tn \times n}$ from the uniform distribution over $(\mathbb{F}^{tn \times n}, \mathbb{F}^{tn})$, contradicting our assumption. $\square$

LEMMA 10.13 (CORRECTNESS). *For the parameters as defined in Construction 10.11, for every field* $\mathbb{F}$*, it holds that*

$$\Pr\left[\mathsf{Dec}_n^{\mathbb{F}}\left(\mathsf{Enc}_n^{\mathbb{F}}(x, s), s\right) = x\right] \geq 1 - \text{neg}(n).$$

PROOF. Decryption fails only if the fraction of noisy coordinates in the vector $e \xleftarrow{R} \chi_p^\ell$ is larger than $\frac{1}{2}$. Since the constant p is strictly smaller than $\frac{1}{2}$, by a Chernoff bound, the latter event happens with probability at most $2^{-\Omega(n)}$. $\square$

COROLLARY 10.14. *Let* $p \in (0, \frac{1}{2})$ *be a constant, and assume* $\mathsf{RLC}(n, \ell, p)$ *holds for every polynomial* $\ell(n)$*; then there exists an IND-CPA secure symmetric encryption scheme in the arithmetic model.*

10.4. Direct Construction of Public-Key Encryption

In this section, we present a direct construction of an arithmetic public-key encryption scheme based on a variant of Alekhnovich's (binary) cryptosystem [Alekhnovich 2003]. Again, the underlying RLC assumption is worse than the one used in Corollary 10.10, and the scheme is given in order to show that LPN-based encryption arithmetizes. We begin with a scheme that only offers a weak form of correctness and later amplify it to a full-fledged scheme via standard techniques.

CONSTRUCTION 10.15 (CPA-SECURE PUBLIC KEY ENCRYPTION SCHEME). *For a security parameter n, let* $p = \frac{1}{2\sqrt{n}}$, $\ell = 2n$.

—KGen_n:
  —*Sample* $A \xleftarrow{R} \mathbb{F}^{\ell\times n}$, $s \xleftarrow{R} \mathbb{F}^n$ *and a vector of field elements* $e \xleftarrow{R} \chi_p^\ell(\mathbb{F})$.
  —*Compute* $b = As + e$. *Let* $M = (b|A)$, *where* $|$ *denotes column concatenation.*
  —*Sample a random* $\ell \times (\ell - n - 1)$ *matrix* B, *which spans* $\ker(M^T)$,[21] *that is,* $MB = 0^{(n+1)\times(\ell-n-1)}$.
  —*Let:* $\mathsf{pk} = B \in \mathbb{F}^{\ell\times(n-1)}$, $\mathsf{sk} = e \in \mathbb{F}^\ell$.
—$\mathsf{Enc}(\mathsf{pk}, x)$: *Sample* $s' \xleftarrow{R} \mathbb{F}^{n-1}$, $e' \xleftarrow{R} \chi_p^\ell$. *Output:* $c = Bs' + x^\ell + e'$, *and recall that* x^ℓ *is an* ℓ *long vector whose entries are all equal to* x.
—$\mathsf{Dec}(\mathsf{sk}, c)$: *If* $\sum_{i=1}^{\ell} e_i = 0$, *output* Fail. *Otherwise, compute* $x' = \frac{e^T\cdot c}{\sum_{i=1}^{\ell} e_i}$ *and output* x'.

LEMMA 10.16 (CORRECTNESS). *For every field* $\mathbb{F}$, *and for all* $x \in \mathbb{F}$ *and all sufficiently large ns:*

$$\Pr_{(\mathsf{pk},\mathsf{sk})\xleftarrow{R}\mathsf{KGen}_n}[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, x)) = \mathsf{Fail}] \leq \frac{1}{|\mathbb{F}|} + \mathrm{neg}(n). \tag{22}$$

Conditioned on not failing, the probability of successful decryption is

$$\Pr_{(\mathsf{pk},\mathsf{sk})\xleftarrow{R}\mathsf{KGen}_n}[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, x)) = x] > 0.51, \tag{23}$$

for all sufficiently large ns.

PROOF. We condition on the event that the Hamming weight of e is larger than 1 and smaller than $1.1p\ell$, which, by a multiplicative Chernoff bound, happens with probability $1 - 2^{-\Omega(p\ell)} = \mathrm{neg}(n)$.

Equation (22) now follows by noting that $\sum e_i$ is uniformly distributed over $\mathbb{F}$ and so it equals zero with probability at most $1/|\mathbb{F}|$.

We now turn to show Equation (23). Observe that the output of the decryption algorithm x' can be written as

$$\frac{e^T \cdot Bs' + x\sum e_i + e^T e}{\sum e_i} = x + \frac{e^T e}{\sum e_i},$$

where the equality follows from the fact that $e^T \in \mathrm{Im}(M^T)$ and therefore $e^T B = 0^{n-1}$. We conclude that decryption is correct if $e^T e' = 0$. Let us fix some vector e. Denoting the Hamming weight of e by $|e|$, it holds that

$$\Pr_{e'}[e^T e' = 0] > (1-p)^{|e|} > (1-p)^{1.1p\ell} > \exp(-1.1/2) - o(1) > 0.51$$

for all sufficiently large ns. □

Denote the previous scheme as PKE. In order to prove security, we define a variant of PKE, denoted PKE′, in which the public key $B \xleftarrow{R} \mathbb{F}^{\ell\times(n-1)}$, and Enc is defined as in PKE. Although we did not define a decryption algorithm for PKE′, its CPA security is still

[21]Such a B can be sampled by first finding any matrix that spans $\ker(M^T)$ and then multiplying it by a random invertible matrix.

well defined. Our proof will now proceed as follows. Under the RLC assumption, we will first prove that PKE' is semantically secure (Lemma 10.17). We will then show that for every efficient field family $\mathbb{F} = \{\mathbb{F}_n\}$, it holds that $\mathsf{pk} \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$ is computationally indistinguishable from $\mathsf{pk} \xleftarrow{R} \mathbb{F}^{\ell\times(n-1)}$ (Claim 10.18) and conclude that Construction 10.15 is semantically secure (Lemma 10.19).

CLAIM 10.17. *Suppose that* $\mathsf{RLC}(n-1, \ell, p)$ *holds for* $\ell = 2n$ *and* $p = \frac{1}{2\sqrt{n}}$*; then* PKE' *is semantically secure.*

PROOF. Assume toward a contradiction that an efficient adversary $\mathcal{A}$ wins the IND-CPA-game against the security of PKE' with probability $1/2 + \delta(n)$ for some nonnegligible function δ. Since we are in the public-key setting, we may assume, without loss of generality, that the adversary does not issue encryption queries. We define a new adversary $\mathcal{B}$ against the $\mathsf{RLC}(n-1, 2n, \frac{1}{2\sqrt{n}})$ assumption in the following way:

Adversary $\mathcal{B}$:

—Initiate $\mathcal{A}$ and get $\mathbb{F}$ from it. Send the same $\mathbb{F}$ to the challenger.
—Receive from the challenger $(B, v) \in (\mathbb{F}^{\ell\times(n-1)}, \mathbb{F}^{\ell})$.
—Send the public key B to $\mathcal{A}$.
—When $\mathcal{A}$ sends the challenge x_0, x_1:
  —Sample $b \in \{0, 1\}$.
  —Send $(v + x_i^{\ell})$ to $\mathcal{A}$.
—Let b' denote $\mathcal{A}$'s output. If $b' = b$ (i.e., $\mathcal{A}$ wins), return 1, or else return 0.

When $(B, v) \xleftarrow{R} (\mathbb{F}^{\ell\times(n-1)}, \mathbb{F}^{\ell})$, then the message x_b is information-theoretically hidden and so $\mathcal{A}$'s winning probability is exactly 1/2 and

$$\Pr[\mathcal{B} \text{ outputs } 0 \mid \text{The challenger uses uniform samples}] = \frac{1}{2}.$$

On the other hand, when $(B, v) \xleftarrow{R} \mathcal{D}_{\mathbb{F},p}^{\ell\times(n-1)}$, the vector $v + x_i^{\ell}$ is identically distributed as $\mathsf{Enc}(B, x_i)$, and so $\mathcal{A}$ guesses b with probability $1/2 + \delta(n)$. It follows that

$$\Pr[\mathcal{B} \text{ outputs } 1 \mid \text{The challenger uses RLC-samples}] = \frac{1}{2} + \delta(n).$$

Overall, $\mathcal{B}$ wins the distinguishing game of the RLC assumption (defined in Assumption 8.1) with probability $\frac{1}{2} + \frac{\delta(n)}{2}$, which contradicts the $\mathsf{RLC}(n-1, 2n, \frac{1}{2\sqrt{n}})$ assumption. □

We next show that the public keys of PKE and PKE' are computationally indistinguishable.

CLAIM 10.18 ($\mathsf{PKE} \overset{c}{\approx} \mathsf{PKE}'$). *Suppose that* $\mathsf{RLC}(n, \ell = 2n, p = \frac{1}{2\sqrt{n}})$ *holds; then, for every efficient adversary* $\mathcal{A}$*, the probability of winning the following game is at most* $\frac{1}{2} + \mathrm{neg}(n)$*:*

IND-Game(1^n)*:*

—$\mathcal{A}$ *receives* 1^n*, chooses a field* $\mathbb{F}$*, and sends* $\mathbb{F}$ *to the challenger.*
—*The challenger samples* $(B_0, e) \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$ *as defined in Construction 10.15, and* $B_1 \xleftarrow{R} \mathbb{F}^{\ell\times(n-1)}$.

Table III. The Hybrid Distributions

KGen_n	$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathbb{F}^{\ell\times(n-1)}$
$A \xleftarrow{R} \mathbb{F}^{\ell\times n}, s \xleftarrow{R} \mathbb{F}^n, e \xleftarrow{R} \chi_p^\ell, b = As + e$	$b \xleftarrow{R} \mathbb{F}^\ell$		
$M = (b\|A)$		$M \xleftarrow{R} \mathcal{U}^{\ell\times n+1}_{\mathrm{rk}=n+1}$	
$B \xleftarrow{R}$ a random matrix that spans $\ker(M^T)$			$B \xleftarrow{R} \mathbb{F}^{\ell\times(n-1)}$
output: B			

$\mathbb{F}$ is the field family chosen by $\mathcal{A}(1^n)$. For readability, only the modifications with respect to the previous hybrid distribution appear.

—*The challenger samples* $b \in \{0, 1\}$ *and sends* B_b *to* $\mathcal{A}$.
—$\mathcal{A}$ *outputs* b' *and wins if* $b = b'$.

Proof. Let $\mathcal{A}$ be an efficient distinguisher and let $\mathbb{F} = \{\mathbb{F}_n\}$ be the efficient field family it outputs. We use a hybrid argument to show that $\mathcal{A}$ wins the game with probability no more than $\frac{1}{2} + \mathrm{neg}(n)$. Let $\mathcal{U}^{\ell\times n}_{\mathrm{rk}=n}$ denote the uniform distribution over the set of all matrices in $\mathbb{F}^{\ell\times n}$ with full rank. In Table III, we define three variants of the key-generation algorithm, where the first hybrid corresponds to the real KGen and the last hybrid corresponds to the case where the key is random. We show that each pair of neighboring hybrids is indistinguishable.

—$\mathsf{KGen}_n \overset{c}{\approx} \mathcal{H}_1$: A distinguisher between the two distributions immediately implies an adversary that wins the IND-game of $\mathsf{RLC}(n, 2n, \frac{1}{2\sqrt{n}})$ over the same field family with the same advantage.
—$\mathcal{H}_1 \overset{s}{\approx} \mathcal{H}_2$: In $\mathcal{H}_1$, we have that $M \xleftarrow{R} \mathbb{F}^{\ell\times(n+1)}$. It is not hard to see that such a random matrix has full rank except with probability $n|\mathbb{F}|^{-\ell+n} \leq \mathrm{neg}(n)$.
—$\mathcal{H}_2 \overset{s}{\approx} \mathbb{F}^{\ell\times(n-1)}$: Clearly the kernel of $M \xleftarrow{R} \mathcal{U}^{\ell\times n+1}_{\mathrm{rk}=n+1}$ is a random subspace of $\mathbb{F}^\ell$ of dimension $\ell - n - 1 = n - 1$. A random matrix that spans it is a uniformly distributed full-rank matrix in $\mathbb{F}^{\ell\times(n-1)}$, and so, by the previous argument, is statistically close to a uniform matrix over $\mathbb{F}^{\ell\times(n-1)}$.

It follows that for the field family $\mathbb{F}$, the ensemble KGen_n is computationally indistinguishable from the uniform distribution over $\mathbb{F}_n^{\ell\times(n-1)}$; hence, $\mathcal{A}$ can win the distinguishing game with probability no more than $\frac{1}{2} + \mathrm{neg}(n)$. $\square$

We can now prove the security of Construction 10.15.

Lemma 10.19 (Security). *Suppose that* $\mathsf{RLC}(n-1, \ell, p)$ *holds for* $\ell = 2n$ *and* $p = \frac{1}{2\sqrt{n}}$; *then Construction 10.15 is CPA secure.*

Proof. Assume toward contradiction that adversary $\mathcal{A}$ wins the IND-CPA-game against Construction 10.15 with probability $\frac{1}{2} + \delta(n)$ for a nonnegligible δ. Consider the following adversary $\mathcal{B}$, which distinguishes a random Alekhnovich pk from a random matrix.

Adversary $\mathcal{B}$:

—Initiate $\mathcal{A}$ and get $\mathbb{F}$ from it. Send the same $\mathbb{F}$ to the challenger.
—Receive from the challenger $B \in \mathbb{F}^{\ell\times(n-1)}$.
—Send the public key B to $\mathcal{A}$.
—When $\mathcal{A}$ requests an encryption of m:
—Send $\mathsf{Enc}(B, x)$ to $\mathcal{A}$.

—For the challenge x_0, x_1:
  —Sample $b \in \{0, 1\}$.
  —Send $\mathsf{Enc}(B, x_b)$ to $\mathcal{A}$.
—If $\mathcal{A}$ wins return 0; otherwise, return 1.

By Lemma 10.17, if $B \xleftarrow{R} \mathbb{F}^{\ell\times(n-1)}$, then $\mathcal{A}$ wins (and $\mathcal{B}$ looses) with probability no more than $\frac{1}{2} + \mathrm{neg}(n)$. By assumption, when $B \xleftarrow{R} \mathsf{KGen}_n^{\mathbb{F}}$, the adversary $\mathcal{A}$ wins (and $\mathcal{B}$ wins) with probability $\frac{1}{2} + \delta(n)$ for a nonnegligible δ. Hence, overall, $\mathcal{B}$ wins the IND-game of Claim 10.18 with probability $\frac{1}{2} + \frac{\delta(n)-\mathrm{neg}(n)}{2}$, a nonnegligible advantage. To prove the lemma (via Claim 10.18), it suffices to show that $\mathsf{RLC}(n, \ell = 2n, p = \frac{1}{2\sqrt{n}})$ is hard.

Recall that we only assumed the hardness of $\mathsf{RLC}(n-1, 2n, \frac{1}{2\sqrt{n}})$ and not $\mathsf{RLC}(n, 2n, \frac{1}{2\sqrt{n}})$; however, this is not a real issue since $\mathsf{RLC}(n-1, \ell, p)$ implies $\mathsf{RLC}(n, \ell, p)$, as argued in Proposition 8.2. □

Building on Lemmas 10.16 and 10.19, we can use amplification to create a PKE scheme with overwhelming success probability in the following way. We repeat the previous construction, each time with fresh keys, polynomially many times. To decrypt, we decrypt (via the erroneous decryption algorithm) each copy of the ciphertext and take the majority. Since Fail occurs over any field with probability no more than 1/2, with overwhelming probability, there will be polynomially many nonfailed decryptions, among which, again by Chernoff's inequality, with overwhelming probability a majority of the decryptions will output the correct value. Security will not be harmed as each execution uses fresh randomness.

COROLLARY 10.20. *Assuming* $\mathsf{RLC}(n-1, \ell, p)$ *holds for* $\ell = 2n$ *and* $p = \frac{1}{2\sqrt{n}}$, *then there exists a CPA-secure public-key encryption scheme in the arithmetic model.*

Again, we emphasize that Corollary 10.10 provides arithmetic PKE under a weaker (and therefore better) assumption.

11. COMMITMENTS

In this section, we describe two constructions for statistically binding string commitment schemes under RLC assumptions. In Section 11.1, we present a noninteractive construction in the *Common Reference String* model where we assume that a trusted party honestly generates the keys by invoking the key-generation algorithm. The advantage of this construction is that it carries some homomorphic properties. In Section 11.2, we give a construction in the standard model by showing that the PRG-based construction of Naor [1991] arithmetizes, and by plugging in an arithmetic PRG based on the RLC assumption.

11.1. Noninteractive Statistically Binding String Commitment

Noninteractive statistically binding string commitment schemes based on the hardness of learning parity with noise were given in Applebaum et al. [2010] and Jain et al. [2012]. Using the RLC assumption, we generalize their schemes to the arithmetic setting and build a commitment scheme for tuples of field elements.

Our construction is in the CRS model where we assume that KGen is executed by some trusted party. For the definition of a commitment scheme, see Definition 4.4.

CONSTRUCTION 11.1 (STATISTICALLY BINDING STRING COMMITMENT). *The commitment scheme is parameterized by the security parameter* $n \in \mathbb{N}$, *a noise parameter* $p \in (0, 1/8)$, *and a length parameter* $\ell = cn$, *where* $c = \lceil \frac{2}{1-H_2(4p)-\varepsilon} \rceil$, *and* $\varepsilon \in (0, 1 - H_2(4p))$ *is an* *arbitrarily small constant. Set $w = \lfloor \ell p \rfloor$. The algorithms of the commitment scheme are as follows:*

—$\mathsf{KGen}_n^{\mathbb{F}}$*: The public commitment key consists of the matrix $A = (A'|A'') \in \mathbb{F}^{\ell\times(2n)}$, where $A' \xleftarrow{R} \mathbb{F}^{\ell\times n}$ and $A'' \xleftarrow{R} \mathbb{F}^{\ell\times n}$.*

—$\mathsf{Com}_n^{\mathbb{F}}(A, x)$*: To commit to a message $x \in \mathbb{F}^n$ using the public key A, the sender samples $r \xleftarrow{R} \mathbb{F}^n$ and $e \xleftarrow{R} \chi_p^{\ell}$, and computes the commitment $c = A\cdot(r|x)+e$. The decommitment of the commitment is the pair (x, r).*

—$\mathsf{Ver}_n^{\mathbb{F}}(A, c, (x', r'))$*: Given a public key A, a commitment c, and a decommitment (x', r'), the receiver computes $e' = c - A\cdot(r'|x')$ and outputs 1 if and only if $w(e') \le 2w$.*

The constant c was chosen to satisfy the following inequality for every integer $q \ge 2$:

$$c = \left\lceil \frac{2}{1 - H_2(4p) - \varepsilon} \right\rceil \ge \left\lceil \frac{2}{1 - H_q(4p) - \varepsilon} \right\rceil > 2, \tag{24}$$

where $H_q(\alpha) := -\alpha \log_q(\alpha) - (1-\alpha)\log_q(1-\alpha)$ denotes the q-ary entropy function. To see that Equation (24) holds, recall that $p \in (0, 1/8)$, and so $4p \in (0, 1/2)$. Since for every $\alpha \in (0, 1/2)$ we have that $H_2(\alpha) \ge H_q(\alpha) > 0$, the equation follows.

We turn now to establishing correctness, that is, that in an honest execution, the verifier accepts with overwhelming probability. Indeed, if the sampled vector $e \xleftarrow{R} \chi_p^{\ell}$ is such that $w(e) \le 2w$, then the verification succeeds with probability 1. By Chernoff's inequality for $p \in (0, 1/8)$ and $\ell = cn$ (for any constant c), the probability that the weight of e exceeds $2w$ is negligible in n. We now turn to security.

LEMMA 11.2. *Suppose that the* $\mathsf{RLC}(n, \ell, p)$ *assumption holds for some constant $p \in (0, 1/8)$, and for $\ell = \lceil \frac{2n}{1-H_2(4p)-\varepsilon} \rceil$. Then Construction 11.1 is a statistically binding and computationally hiding commitment scheme.*

PROOF. Let us call A *good* if it generates a code whose distance is larger than $4\ell p$. It is well known (cf. Venkatesan Guruswami and Sudan [2014, Chapter 4]) that a randomly chosen generating matrix $M \xleftarrow{R} \mathbb{F}^{\ell\times n}$ spans an error-correcting code with distance of at least $\delta\ell$ with probability of at least $|\mathbb{F}|^{-\varepsilon\ell}$ for any $\delta \in (0, 1 - \frac{1}{|\mathbb{F}|})$ and $\ell \ge \frac{n}{1-H_{|\mathbb{F}|}(\delta)-\varepsilon}$. Plugging in $\delta = 4p$ and $\ell = cn$ for c that satisfies Equation (24), we conclude that A is good with overwhelming probability.

Let us prove that, conditioned on A being good, the protocol is statistically binding. Assume toward contradiction that $x_i, r_i, i = 1, 2$ are two different decommitments for the same commitment c. Then $e_i = c - A(r_i|x_i)$ has a weight of no more than $2w$, and so $e_1 - e_2 = A(r_1 - r_2|x_1 - x_2)$ is a codeword of weight less than $4w$, in contradiction to our hypothesis regarding the distance of A.

Finally, we prove that the commitment scheme is computationally hiding. Assume that an adversary $\mathcal{A}$ breaks the hiding property and wins the indistinguishability game with some nonnegligible advantage δ. Consider the following adversary against the RLC assumption:

Adversary $\mathcal{B}(1^n)$:

—Initiate $\mathcal{A}(1^n)$ and get $\mathbb{F}$ from it. Send the same $\mathbb{F}$ to the challenger.

—Receive from the challenger $(A', u) \in (\mathbb{F}^{\ell\times n}, \mathbb{F}^{\ell})$.

—Sample $A'' \xleftarrow{R} \mathbb{F}^{\ell\times n}$ and send $\mathsf{pk} = (A'|A'')$ to the adversary $\mathcal{A}$.

—Receive from $\mathcal{A}$ a pair $(x_0, x_1) \in (\mathbb{F}^n, \mathbb{F}^n)$.

—Sample $b \in \{0, 1\}$, compute $c = u + A'' \cdot x_b$, and send c to $\mathcal{A}$.
—Return 0 if $\mathcal{A}$ wins (i.e., $\mathcal{A}$ returns $b' = b$), and return 1 otherwise.

Observe that if $(A', u) \xleftarrow{R} (\mathbb{F}^{\ell\times n}, \mathbb{F}^{\ell})$, then $c \xleftarrow{R} \mathbb{F}^{\ell}$ and $\mathcal{A}$ wins with probability exactly 1/2, and hence $\mathcal{B}$ wins with probability exactly 1/2. On the other hand, when $(A', u) \xleftarrow{R} \mathcal{D}^{\ell\times n}_{\mathbb{F},p}$, then $(\mathsf{pk}, c) \stackrel{i}{\equiv} (\mathsf{pk}, \mathsf{Com}^{\mathbb{F}}_{\mathsf{pk}}(x_b))$, and by assumption, $\mathcal{A}$ wins (hence $\mathcal{B}$ wins) with probability $1/2 + \delta$. Overall, $\mathcal{B}$ wins the IND-game against the RLC assumption with the nonnegligible advantage $\delta/2$, in contradiction to our hypothesis. □

COROLLARY 11.3. *Assuming the* $\mathsf{RLC}(n, \ell, p)$ *assumption holds for some constant* $p \in (0, 1/8)$, *and for* $\ell = \lceil \frac{2n}{1-H_2(4p)-\varepsilon} \rceil$, *there exists an arithmetic commitment scheme in CRS model.*

11.2. CRS Free Commitment Protocol

Our next commitment scheme is an arithmetic analog of Naor's commitments [Naor 1991] and does not require a common reference string. Syntactically, we still consider a noninteractive commitment function parameterized by a public key that is generated by a key-generation algorithm. However, the hiding property should hold for every (adversarially chosen) public key. This gives rise to a two-message commitment protocol in the standard model (no CRS) in which the receiver chooses the public key pk. The construction is based on an *arithmetic pseudorandom generator* APRG_n, which stretches n inputs to $3n$ outputs. In the following, we will write $\mathsf{APRG}_n(s)$ to denote explicitly the output of $\mathsf{APRG}_n(s)$ on a seed s. Note that the total length of s is n and it may consist of both random field elements and random bits.

CONSTRUCTION 11.4 (CRS FREE STATISTICALLY BINDING COMMITMENT PROTOCOL). *The commitment scheme is parameterized by a security parameter* $n \in \mathbb{N}$. *Let* APRG *be any arithmetic PRG with expansion factor 3. Define the protocol as follows:*

—$\mathsf{KGen}^{\mathbb{F}}_n$: *The receiver publishes the public key* $r \xleftarrow{R} \mathbb{F}^{3n}$.
—$\mathsf{Com}^{\mathbb{F}}_n(r, x)$: *To commit to a message* $x \in \mathbb{F}$ *using the public key* r, *the sender samples a seed* s *for the APRG and computes* $c = \mathsf{APRG}_n(s) + x \cdot r$, *where* $\cdot$ *stands for scalar multiplication. The decommitment is the pair* (s, x).
—$\mathsf{Ver}^{\mathbb{F}}_n(r, c, (s', x'))$: *Given a commitment* c, *a public key* r, *and a decommitment* (s', x'), *the receiver outputs* 1 *if and only if* $c = \mathsf{APRG}(s') + x' \cdot r$.

We now prove that this protocol indeed realizes a commitment scheme.

LEMMA 11.5. *Assuming the pseudorandomness of* APRG, *the scheme is computationally hiding. Furthermore, this holds even if the public key is chosen adversarially.*

PROOF. Assume that an efficient malicious receiver $\mathcal{R}$ can break the hiding property of the scheme. $\mathcal{R}$ chooses the field $\mathbb{F}$ and a public key r for which it wins the distinguishing game in Definition 4.4 with probability $\frac{1}{2} + \delta(n)$ for some nonnegligible δ. Consider the following adversary against the security of the APRG:

Adversary $\mathcal{A}$:

(1) Invoke $\mathcal{R}$ and get $\mathbb{F}$ form it.
(2) Get $r \in \mathbb{F}^{3n}$ and $x_0, x_1 \in \mathbb{F}$ from $\mathcal{R}$.
(3) Initialize the distinguishing game against the APRG with the field $\mathbb{F}$:
  (a) Get $u \in \mathbb{F}^{3n}$ from a challenger.
  (b) Sample $b \xleftarrow{R} \{0, 1\}$ and send $\mathcal{R}$ the commitment $c = u + x_b \cdot r$.
  (c) If $\mathcal{R}$ returns $b' = b$, output 1. Otherwise, output 0.

By assumption, when u is a random output of the APRG, c is distributed identically to an honestly generated commitment and $\mathcal{R}$ must win with probability $\frac{1}{2} + \delta(n)$ for some nonnegligible δ. Hence, in this case, $\mathcal{A}$ returns 1 and wins also with the probability of $\frac{1}{2} + \delta(n)$. However, when $u \xleftarrow{R} \mathbb{F}^n$, c hides b information theoretically, and therefore $\mathcal{R}$ wins with probability exactly 1/2, and $\mathcal{A}$ returns 0 and wins with probability exactly 1/2. Overall, $\mathcal{A}$ wins with probability $\frac{1}{2} + \frac{\delta(n)}{2}$, in contradiction to the security of the APRG. □

LEMMA 11.6. *With overwhelming probability over $r \xleftarrow{R} \mathbb{F}^{3n}$, the scheme is statistically binding.*

PROOF. Call a public key $r \in \mathbb{F}^{3n}$ *ambiguous* if there exist a pair of messages $x_0 \neq x_1$, and a pair of seeds s_0 and s_1 for which $\mathsf{APRG}(s_0) + x_0 r = \mathsf{APRG}(s_1) + x_1 r$. This happens if and only if $r = \frac{\mathsf{PRG}(s_0) - \mathsf{APRG}(s_1)}{x_1 - x_0}$; however, the right-hand side consists of no more than $|\mathbb{F}|^{2n+1}$ possible vectors, and so the probability that a random r is ambiguous is no more than $|\mathbb{F}|^{2n+1}/|\mathbb{F}|^{3n} = |\mathbb{F}|^{-n+1}$, which is negligible. □

Since an arithmetic PRG with expansion factor of 3 can be based on any APRG (Lemma 9.2), and since the latter can be based on the RLC assumption (Theorem 9.3), we obtain the following corollary.

COROLLARY 11.7. *Assuming an arithmetic PRG, there exists an arithmetic commitment protocol that does not require a common reference string. Specifically, if $\mathsf{RLC}(n, \ell = \frac{n}{1-p-\varepsilon}, p)$ holds for some constants $p \in (0, 1), \varepsilon > 0$, then there exists an arithmetic commitment protocol that does not require a common reference string.*

12. SECURE COMPUTATION

In this section, we show how to securely compute any two-party arithmetic functionality. We start with simple arithmetic functionalities such as $\binom{2}{1}$-Arithmetic Oblivious Transfer (Section 12.1) and Oblivious Linear Evaluation (Section 12.2), and eventually (Section 12.3) show how to privately compute any two-party functionality that is described by an arithmetic circuit over addition and multiplication gates only. (The results extend to the multiparty setting in a straightforward way.) We will mostly focus on the semihonest setting. An adaptation to the malicious model is sketched in Section 12.4.

12.1. $\binom{2}{1}$-Arithmetic Oblivious Transfer

We define $\binom{2}{1}$ Arithmetic Oblivious Transfer via the following partial functionality: the receiver's input is a selection bit $x \in \{0, 1\}$ and security parameter 1^n, and the sender's input is a pair of field elements $z_0, z_1 \in \mathbb{F}$ and security parameter 1^n. The functionality delivers to the receiver the value z_x (which can be written arithmetically as $x \cdot z_1 + (1 - x) z_2$). The sender receives no output (an empty string). The more standard binary variant of this functionality (denoted $\binom{2}{1}$-OT) corresponds to the case where the sender's inputs z_0, z_1 are binary strings (of poly(n) length).

We begin by showing how to use ABE to privately reduce $\binom{2}{1}$-AOT to standard binary $\binom{2}{1}$-OT. The construction is similar to the standard transformation from OT over short strings to OT over long strings.

CONSTRUCTION 12.1. *Let* OT *be any binary (string) oblivious transfer protocol, and let $\mathcal{E} = (\mathsf{ABGen}, \mathsf{ABEnc}, \mathsf{ABDec})$ be any arithmetic/binary encryption scheme. Let $z_0, z_1 \in \mathbb{F}$*

be the input of the sender and $x \in \{0, 1\}$ the input of the receiver. Consider the following scheme:

—*Sender samples binary keys* $\mathsf{sk}_0, \mathsf{sk}_1 \xleftarrow{R} \mathsf{ABGen}_n$.
—*Both parties run the binary* OT *protocol with the input* $(x, (\mathsf{sk}_0, \mathsf{sk}_1))$ *(recall that* sk_i *is a binary string).*
—*Sender sends to the receiver the vector (over* $\mathbb{F}$*)* $c_i \xleftarrow{R} \mathsf{ABEnc}_n^{\mathbb{F}}(\mathsf{sk}_i, z_i)$ *for* $i \in \{0, 1\}$.
—*Receiver outputs* $z_x = \mathsf{ABDec}_n^{\mathbb{F}}(\mathsf{sk}_x, c_x)$. *In case decryption fails, it outputs* 1.

THEOREM 12.2. *Assuming that $\mathcal{E}$ is an ABE, Construction 12.1 privately reduces $\binom{2}{1}$-AOT to standard binary $\binom{2}{1}$-OT.*

In fact, it can be shown that, if the underlying binary OT is secure in the malicious setting, then so is the resulting $\binom{2}{1}$-AOT. (See Section 12.4.1.)

PROOF. The correctness of Construction 12.1 follows immediately from the correctness of the binary OT and the correctness of the ABE. The simulator for the sender is trivial, as it gets no message. The simulator $\mathsf{Sim}_2(x, z_x)$ for the receiver generates a view that corresponds to messages sent by a Sender whose xth input is z_x and its other input z_{1-x} is set to zero. Formally, $\mathsf{Sim}_2(x, z_x)$ samples $\mathsf{sk}_0, \mathsf{sk}_1 \xleftarrow{R} \mathsf{ABGen}_n$; computes the ciphertexts $c_i \xleftarrow{R} \mathsf{ABEnc}_n^{\mathbb{F}}(\mathsf{sk}_i, z_i)$, where $z_{1-x} = 0$; and outputs the tuple $(x, \mathsf{sk}_x, c_0, c_1)$. The security of the ABE guarantees that the simulated view is indistinguishable from the real view. □

Combining the construction with Corollary 10.4, we derive the following corollary.

COROLLARY 12.3. *Assume that for some constant $p \in (0, 1)$ and constant $\varepsilon > 0$ the* $\mathsf{RLC}(n, \ell = \frac{(1+\varepsilon)n}{1-p}, p)$ *assumption holds. Then, assuming the existence of a binary Oblivious Transfer protocol, there exists an arithmetic $\binom{2}{1}$-AOT protocol.*

The latter assumption (existence of binary Oblivious Transfer protocol) is necessary as an arithmetic $\binom{2}{1}$-AOT protocol immediately implies the existence of binary Oblivious Transfer.

12.1.1. Alternative Construction-Based Alekhnovich's PKE. An alternative construction of $\binom{2}{1}$-AOT can be established by observing that the public key in Alekhnovich's encryption scheme is pseudorandom.

CONSTRUCTION 12.4. *The protocol is described in Figure 2.* $\mathsf{Gen}, \mathsf{Dec}, \mathsf{Enc}$ *are the arithmetic circuits of the Alekhnovich PKE scheme, as defined in Construction 10.15.*

Correctness of the scheme follows from the correctness of the underlying PKE scheme.

LEMMA 12.5 (PRIVACY). *Suppose that the* $\mathsf{RLC}(n-1, 2n, \frac{1}{2\sqrt{n}})$ *assumption holds; then Construction 12.4 privately realizes the $\binom{2}{1}$-AOT functionality.*

PROOF. Correctness follows from the correctness of Alekhnovich's PKE (Lemma 10.16). The simulator of the sender $\mathsf{Sim}_1(z_0, z_1)$ simply outputs a pair of random matrices $M_0, M_1 \xleftarrow{R} \mathbb{F}^{\ell \times n-1}$ (together with its inputs and its internal randomness). The pseudorandomness of public keys (as established in Claim 10.18) shows that the simulated view is indistinguishable from the real view. The simulator of the receiver $\mathsf{Sim}_2(x)$ essentially replaces c_{1-x} with an encryption of zero. Formally,

Sender		Receiver
Input: $z_0, z_1 \in \mathbb{F}$		Input: $x \in \{0,1\}$
		$(M_x, e) \xleftarrow{R} \mathsf{KGen}_n$
		$M_{1-x} \xleftarrow{R} \mathbb{F}^{\ell \times n-1}$
	$\xleftarrow{\quad M_0, M_1 \quad}$	
$c_0 \xleftarrow{R} \mathsf{Enc}_n(M_0, z_0)$		
$c_1 \xleftarrow{R} \mathsf{Enc}_n(M_1, z_1)$		
	$\xrightarrow{\quad c_0, c_1 \quad}$	
		$z_x = \mathsf{Dec}_n(c_x, e)$

Fig. 2. $\binom{2}{1}$-AOT from Alekhnovich.

$\mathsf{Sim}_2(x, z_x)$ samples $(M_x, e) \xleftarrow{R} \mathsf{KGen}_n$ and $M_{1-x} \xleftarrow{R} \mathbb{F}^{\ell \times n-1}$ computes $c_x = \mathsf{Enc}_n(M_x, z_x)$ and $c_{1-x} = \mathsf{Enc}_n(M_{1-x}, 0)$ and outputs $(x, M_0, M_1, e, c_0, c_1)$. The semantic security of Alekhnovich's encryption over random keys (Claim 10.17) implies that the simulated view is indistinguishable from the real view. $\square$

Corollary 12.6. *Assuming* $\mathsf{RLC}(n-1, 2n, \frac{1}{2\sqrt{n}})$, *there exists a computationally secure* $\binom{2}{1}$*-AOT protocol in the semihonest model.*

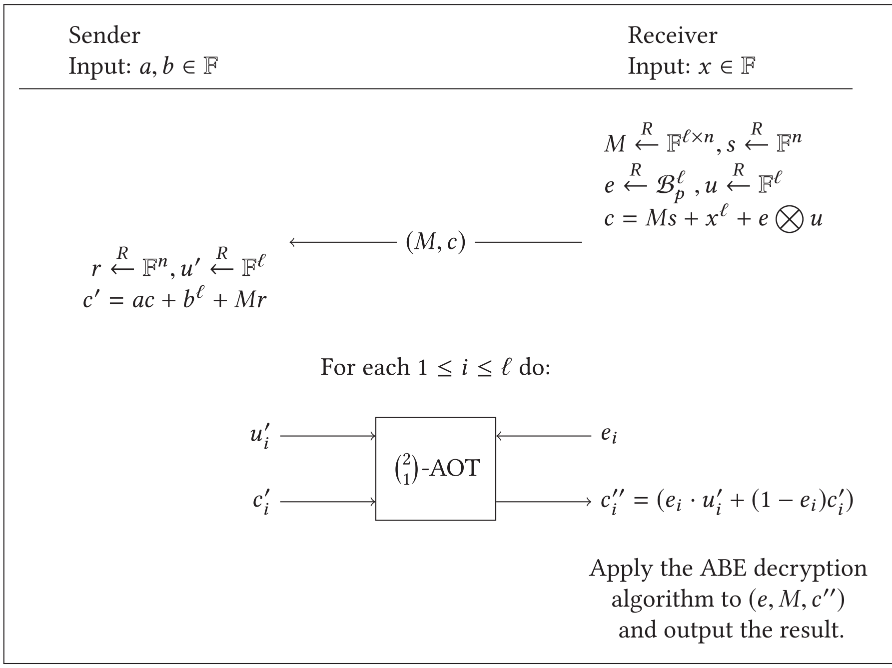
12.2. From $\binom{2}{1}$-AOT to Oblivious Linear Evaluation

The Oblivious Linear Evaluation (OLE) functionality takes a single field element $x \in \mathbb{F}$ from the receiver and a pair of field elements $a, b \in \mathbb{F}$ from the sender and delivers to the receiver the value $ax + b$ (and nothing to the sender). As before, we assume that both parties are also given a common security parameter 1^n.

OLE via (Weakly) Homomorphic Encryption. A natural way to obtain OLE is via the use of a (weak) *homomorphic encryption*. Specifically, let the receiver encrypt $x \in \mathbb{F}$, and let the sender homomorphically modify the ciphertext $c = E_k(x)$ to $c' = E_k(ax + b)$. Since the new ciphertext may leak information on a and b (as is the case in our RLC-based OLE), we will refresh the ciphertext c' via the use of a subprotocol based on $\binom{2}{1}$-AOT. Next we instantiate this approach with the ABE from Construction 10.1. (See also Remark 12.9 for a generalization.)

Concretely, recall that we encrypted a message $x \in \mathbb{F}$ by sampling (M, v) from some RLC distribution ($v = Ms + e$) and used v to pad the (encoded) message x^ℓ. Notice that such an encryption has weak homomorphic properties. Namely, given a ciphertext $(M, c) = \mathsf{Enc}(x)$ and scalars $a, b \in \mathbb{F}$, we can compute new ciphertexts $(M, c' = a \cdot c + b^\ell)$, which form a valid encryption of $ax + b$. While it is easy to rerandomize the vector s (by adding $M \cdot s'$ for some random s'), the noisy coordinates remain correlated with the original noise vector. We remove this correlation by letting the receiver learn only the nonnoisy coordinate using the $\binom{2}{1}$-AOT. The construction is given in Figure 3. A similar approach (under a different abstraction) appears in Ishai et al. [2009].

Theorem 12.7. *Assuming* $\mathsf{RLC}(n, \ell := \frac{(1+\varepsilon)n}{1-p}, p)$ *for some constant* $p \in (0, 1)$ *and constant* $\varepsilon > 0$, *the construction in Figure 3 privately reduces OLE to* $\binom{2}{1}$*-AOT.*



The notation $\bigotimes$ stands for entry-wise product. Recall that $\mathcal{B}_p^\ell$ samples a binary vector which consists of ℓ independent Bernoulli random variables with mean p.

Fig. 3. OLE from $\binom{2}{1}$-AOT.

Proof. First observe that the ith coordinate of c'' satisfies the equality

$$c''_i = (ac + b^\ell + Mr)_i = (M(as + r) + (ax + b)^\ell)_i \qquad \text{for } i : e_i = 0,$$

and $c''_i = u'_i$ for $i : e_i = 1$. Hence, (M, c'') is distributed as a fresh encryption of $(ax + b)$ under the private key e. Furthermore, this ciphertext is statistically independent of s and e. Correctness now follows from the correctness of the ABE, and the view of the receiver can be perfectly simulated (in the $\binom{2}{1}$-AOT hybrid model) by outputting (x, M, s, e, u, c''), where M, s, e, u are sampled as in the real protocol and c'' is a fresh encryption of $z = ax + b$ (under the key e and the matrix M).

The view of the sender can be simulated by (a, b, r, u', M, c), where r, u' are sampled as in the real protocol and (M, c) is a fresh encryption of zero (under a uniform private key). The security of the ABE implies that the simulated view is indistinguishable from the real view. □

Combined with Corollary 12.3, we derive the following corollary.

Corollary 12.8. *Suppose that for some constants $p \in (0, 1)$ and $\varepsilon > 0$, the* $\mathsf{RLC}(n, \ell := \frac{(1+\varepsilon)n}{1-p}, p)$ *assumption holds and that (standard) $\binom{2}{1}$-OT exists. Then, there exists an arithmetic OLE protocol in the semihonest model.*

Remark 12.9 (*Abstraction*). The previous construction can be abstracted by relying on any ABE as follows: (1) (Weakly Homomorphic) Given a fresh ciphertext $\mathsf{Enc}_e(x)$ and scalars $a, b \in \mathbb{F}$, it is possible to obtain a ciphertext c' that decrypts to $ax + b$; (2) (Secure Decryption) The "decryption" functionality f_{Dec} privately reduces to $\binom{2}{1}$-AOT, where f_{Dec} is the two-party functionality that takes a ciphertext c' from a sender and a private key e from a receiver and delivers the value of $\mathsf{Dec}_e(c')$ to the receiver. The construction in Figure 3 follows this outline by designing a private realization for f_{Dec}, which is tailored to our concrete ABE. An alternative, more generic, approach can be based on information-theoretic arithmetic randomized encoding. Specifically, assume that f_{Dec} admits a fully decomposable randomized encoding (as per Definition 4.6) $\hat{f}$ with a decoder B. Then, f_{Dec} privately reduces to $\binom{2}{1}$-AOT via the following protocol. The ciphertext holder will choose randomness for the encoding, will send the part of the encoding $\hat{f}(c', e)$ that depends only on the ciphertext c' in the clear, and, for the ith bit of the private key, use $\binom{2}{1}$-AOT to let the secret-key holder learn the part of the encoding that depends on $e_i \in \{0, 1\}$. (See Ishai and Kushilevitz [2000] and Applebaum et al. [2006] for a security proof of this protocol.) It is shown in Cramer et al. [2003] that any function computed by arithmetic branching program admits (information-theoretic) fully decomposable arithmetic randomized encoding. Hence, the previous approach can be applied to any weakly homomorphic ABE whose decryption algorithm can be implemented by a polynomial-size branching program.

12.3. General Functionalities

In this section, we describe a protocol for privately[22] computing any two-party functionality that is described by an arithmetic circuit over addition and multiplication gates only. We refer to such functionalities as *strictly arithmetic* functionalities. The protocol is an arithmetic version of the well-known (binary) construction of Goldreich et al. [1987] (see also [Goldreich 2004, Chapter 7.3]). Recall the high-level structure of Goldreich et al. [1987]. To compute $f(x, y)$, the parties first secretly share their inputs, and then they propagate through the circuit that computes f at each gate running a subprocedure to compute secret shares of the output of the gate. When reaching the output wires, the parties reveal the relevant secrets so that reconstruction will be possible. The main difficulty is to create a subprocedure for computing the shares of an output of a multiplication gate. We will start by presenting such a protocol.

The two-party functionality MULT takes a pair of shares $(a_1, b_1) \in \mathbb{F}^2$ from the sender and a pair of shares $(a_2, b_2) \in \mathbb{F}^2$ from the receiver. (These shares correspond to the secrets $a = a_1 + a_2$ and $b = b_1 + b_2$.) In addition, the functionality takes a "target" share $c_1 \in \mathbb{F}$ from the sender. The receiver should get a value $c_2 \in \mathbb{F}$ that corresponds, together with c_1, to a sharing of the product ab. Formally, the relation $c_1 + c_2 = (a_1 + a_2) \cdot (b_1 + b_2)$ should hold.[23] It is not hard to privately reduce the MULT functionality to OLE.

Construction 12.10 ($\mathsf{MULT}((a_1, b_1), (a_2, b_2))$). *Consider the following two-party protocol:*

(1) *Inputs: Sender holds $(a_1, b_1, c_1) \in \mathbb{F}^3$; receiver holds $(a_2, b_2) \in \mathbb{F}^2$.*
(2) *Sender samples $r \xleftarrow{R} \mathbb{F}$.*
(3) *The parties engage in an OLE protocol. The sender plays the sender with inputs (a_1, r) and the receiver plays the receiver with the input b_2. Denote the output of the receiver by v_1.*
(4) *The parties engage in an OLE protocol. The sender plays the sender with the values $(b_1, a_1 b_1 - c_1 - r)$ and the receiver plays the receiver with the value a_2. Denote the output of the receiver by $v_2 \in \mathbb{F}$.*
(5) *The receiver outputs the field element $c_2 = v_1 + v_2 + a_2 \cdot b_2$.*

[22]Recall that the term *private* computation refers to the semihonest model, while *secure* computation refers to the malicious model.
[23]This somewhat nonstandard formulation allows us to work with a deterministic functionality.

LEMMA 12.11. *Construction 12.10 privately reduces MULT to OLE.*

PROOF. We first observe the correctness of the protocol. Assuming the correctness of the underlying OLE, $v_1 = a_1 \cdot b_2 + r$ and $v_2 = a_2 \cdot b_1 + a_1 \cdot b_1 - c_1 - r$ and so $c_2 = (a_1 + a_2) \cdot (b_1 + b_2) - c_1$, as required.

We now present the privacy reduction from Construction 12.10 to OLE. For this end, we need to show a simulator for each of the parties' views when the OLE is replaced by an oracle. Such a simulation is straightforward. The sender receives no message, so the simulation is trivial. For the receiver, the simulator gets $((a_2, b_2), c_2 = (a_1 + a_2) \cdot (b_1 + b_2) - c_1)$ and outputs its inputs together with $v_1 \xleftarrow{R} \mathbb{F}$ and $v_2 = c_2 - v_1 - a_2 \cdot b_2$. It is not hard to verify that the simulated view is distributed identically to the receiver's real view. □

We now use Construction 12.10 to securely evaluate general circuits.

CONSTRUCTION 12.12 (PRIVATE CIRCUIT EVALUATION). *Let C be an arithmetic circuit that contains only addition and multiplication gates; some of its input wires are associated with party 1 and the others with party 2. The parties execute the following protocol:*

(*1*) ***Secret-sharing the inputs:*** *Party 1 secret-shares each of its input wires in the following way: for an input wire x, it samples $x_2 \xleftarrow{R} \mathbb{F}$ and sets $x_1 = x - x_2$. It then sends x_2 to party 2. Party 2 does the same for its input wires.*

(*2*) ***Emulating the circuit:*** *Following the circuit evaluation order, the parties use their shares of the wires to compute at each gate a share of the output wire. Specifically, if party i holds shares $a_i \in \mathbb{F}$ and $b_i \in \mathbb{F}$ of two wires that enter some gate, then the share $c_i \in \mathbb{F}$ of the outgoing wire is computed as follows.*
   (*a*) ***Addition gate:*** *Party i locally computes $c_i = a_i + b_i$.*
   (*b*) ***Multiplication gate:*** *The parties run the multiplication sharing protocol of Construction 12.10 with the inputs (a_1, b_1, c_1) and (a_2, b_2), where $c_1 \xleftarrow{R} \mathbb{F}$ and c_2 is the output of the protocol.*

(*3*) ***Recovering the outputs:*** *Once the shares of the output wires are computed, each party sends its share of each output wire to the party with which the output wire is associated. Each party recovers its wires by adding the two shares for the wire.*

Correctness of addition gates is straightforward, correctness of multiplication gates follows the correctness of Construction 12.10, and so the correctness of the entire protocol can be inferred by an inductive argument. We now turn to prove that the privacy of Construction 12.12 reduces to that of Construction 12.10.

PROOF SKETCH. Without loss of generality, we describe the simulator for party 1. The simulator's inputs are all party 1's inputs: denote them by $x_1, \ldots, x_n$, and all party 1's outputs: $y_1, \ldots, y_n$. The simulator works in three steps:

(1) Secret sharing step: Create the secret shares of party 1 as honest party 1. For the shares sent by party 2, sample them uniformly at random.
(2) Circuit evaluation step: Evaluate addition gates as honest party 1. For multiplication gates, sample the output uniformly at random.
(3) Output recovery: For every output wire of party 1, the simulator was given as input the correct output value y. It also has a secret share of the same wire, denoted by a_1, that was computed by it in the previous steps. It outputs the second secret share for this wire as $a_2 = y - a_1$.

We claim that the output of the simulation is distributed identically as the view of party 1. The output of step 1 is identically distributed to the view of party 1 in the true execution. Conditioned on the output of step 1, the output of step 2 is also distributed exactly as the view of party 1. To see this, note that the output c_1 of a multiplication gate is a uniformly random output when c_2 is excluded from the view. Finally, the output generated in step 3 is a deterministic function of the previous distributions, and hence does not affect the distance between the simulation and the real execution. □

THEOREM 12.13. *Any two-party functionality f that can be efficiently described by an arithmetic circuit with only addition and multiplication gates privately reduces to arithmetic OLE. In particular, Construction 12.12 provides such a reduction.*

Combined with Corollary 12.8, we derive the following corollary.

COROLLARY 12.14. *Suppose that for some constants $p \in (0, 1)$ and $\varepsilon > 0$, the $\mathsf{RLC}(n, \ell := \frac{(1+\varepsilon)n}{1-p}, p)$ assumption holds and that (standard) $\binom{2}{1}$-OT exists. Then, any strictly arithmetic two-party functionality can be privately computed in the arithmetic model.*

The corollary can be easily extended to the multiparty setting by using the standard multiparty variant of Construction 12.12.

12.4. The Malicious Model

In this section, we briefly sketch the extension of the previous section to the malicious model.

The following theorem is implicit in Ishai et al. [2009] and is based on the IPS compiler [Ishai et al. 2008] and the arithmetic protocol for secure *multiparty* computation (with information-theoretic security) of Cramer et al. [2003].

THEOREM 12.15. *Assuming the existence of semihonest secure computation for any strictly arithmetic two-party functionality and the existence of $\binom{2}{1}$-AOT with malicious security, any strictly arithmetic two-party functionality can be securely computed (with security against malicious parties) in the arithmetic model.*

In Section 12.4.1, we will show that Construction 12.1 generalizes to the malicious setting.

THEOREM 12.16. *Assuming that $\mathcal{E}$ is an ABE, Construction 12.1 securely reduces (in the malicious setting) $\binom{2}{1}$-AOT to standard binary $\binom{2}{1}$-OT.*

Combining Theorem 12.16 with Corollary 12.14 and Corollary 10.4, we derive the following corollary.

COROLLARY 12.17. *Assuming $\mathsf{RLC}(n, \ell := \frac{(1+\varepsilon)n}{1-p}, p)$ for some constants $p \in (0, 1)$ and $\varepsilon > 0$, and assuming the existence of (standard) $\binom{2}{1}$-OT, any strictly arithmetic two-party functionality can be securely computed (with security against malicious parties) in the arithmetic model.*

As before, this can be extended to the multiparty setting using standard techniques.

12.4.1. Proof of Theorem 12.16. We reduce the malicious security of the $\binom{2}{1}$-AOT to the security of an *Ideal* $\binom{2}{1}$-OT functionality.

Honest receiver. Let A^* be any (malicious) polynomial-time algorithm equipped with auxiliary input aux that plays the role of the sender in the real model. We define a corresponding adversary Sim_{A^*} in the ideal model. Consider the following implementation of Sim_{A^*} (when instantiated with the field $\mathbb{F}$ and the security parameter n):

$\mathsf{Sim}_{A^*}(\mathsf{aux}, z_0, z_1)$:

(1) Invoke $A^*(\mathsf{aux}, z_0, z_1)$.
   (a) Get $\mathsf{sk}'_0, \mathsf{sk}'_1$ that A^* sends to the OT oracle.
   (b) Get c_0, c_1 that A^* sends to the receiver.
(2) For $i \in \{0, 1\}$, compute $z'_i = \mathsf{ABDec}^{\mathbb{F}}_n(\mathsf{sk}'_i, c_i)$. If decryption fails, set $z'_i = 1$.
(3) Send (z'_0, z'_1) to the *trusted party* and receive the empty string ϕ as an answer.
(4) Output $A^*(\mathsf{aux}, z_0, z_1, \phi)$.

It is now straightforward that for every field $\mathbb{F}$, for every input $x \in \{0, 1\}, z_0, z_1 \in \mathbb{F}$, and auxiliary input aux, the real and ideal distributions are identically distributed.

Honest sender. Let B^* be any (malicious) polynomial-time algorithm for the receiver in the real model and let aux be his auxiliary input. We now construct an ideal-world adversary Sim_{B^*} (for the field $\mathbb{F}$ and security parameter n):

$\mathsf{Sim}_{B^*}(\mathsf{aux}, x)$:

(1) Sample $\mathsf{sk}_0, \mathsf{sk}_1 \overset{R}{\leftarrow} \mathsf{ABGen}^{\mathbb{F}}_n$.
(2) Invoke $B^*(\mathsf{aux}, x)$ and let x' be the request that B^* sends to the Binary OT oracle.
(3) Send x' to the *trusted party* and let $z_{x'}$ denote its answer.
(4) Sample $c'_{x'} \overset{R}{\leftarrow} \mathsf{ABEnc}^{\mathbb{F}}_n(\mathsf{sk}_{x'}, z_{x'})$ and $c'_{1-x'} \overset{R}{\leftarrow} \mathsf{ABEnc}^{\mathbb{F}}_n(\mathsf{sk}_{1-x'}, 0)$.
(5) Forward $(\mathsf{sk}_{x'}, c'_0, c'_1)$ to B^* and output its output.

We will show that the simulated view is indistinguishable from the real view. Next we let $B^*_1(\mathsf{aux}, x)$ denote the first message generated by B^* and let $B^*_2(\mathsf{aux}, x, x', \mathsf{sk}'_x, c'_0, c'_1)$ denote the final output of B^*. For every field $\mathbb{F}$, and for every sequence of inputs $x \in \{0, 1\}, z_0, z_1 \in \mathbb{F}$ and auxiliary input aux, we have

$$\begin{aligned}
&\{\mathsf{Real}_{\Pi,(A,B^*(\mathsf{aux}))}(x, (z_0, z_1))\}_{n,x,z_0,z_1,\mathsf{aux}} \\
&\qquad = \{(B^*_2(\mathsf{aux}, x, x', \mathsf{sk}'_x, c'_0, c'_1), \phi),\ \text{where } x' \leftarrow B^*_1(\mathsf{aux}, x)\}_{n,x,z_0,z_1,\mathsf{aux}}. \\
&\qquad \overset{c}{\approx} \{(B^*_2(\mathsf{aux}, x, x', \mathsf{sk}'_x, c_0, c_1), \phi),\ \text{where } x' \leftarrow B^*_1(\mathsf{aux}, x)\}_{n,x,z_0,z_1,\mathsf{aux}} \\
&\qquad\qquad = \{\mathsf{Ideal}_{f,(A,\mathsf{Sim}_{B^*}(\mathsf{aux}))}(x, (z_0, z_1))\}_{n,x,z_0,z_1,\mathsf{aux}},
\end{aligned}$$

where $c_0 = \mathsf{ABEnc}^{\mathbb{F}}_n(\mathsf{sk}_0, z_0)$, $c_1 = \mathsf{ABEnc}^{\mathbb{F}}_n(\mathsf{sk}_1, z_1)$, and ϕ denotes the empty string. Computational indistinguishability follows from the security of the *ABE* scheme, since any adversary that distinguishes the two distributions immediately translates to an adversary that distinguishes an encryption of 0 from an encryption of $z_{1-x'}$. $\square$

ACKNOWLEDGMENTS

We thank the anonymous referees of ITCS 2015 and JACM for their useful comments, and Yuval Ishai for valuable discussions.

REFERENCES

Scott Aaronson and Avi Wigderson. 2008. Algebrization: A new barrier in complexity theory. In *40th Annual ACM Symposium on Theory of Computing*, Richard E. Ladner and Cynthia Dwork (Eds.). ACM Press, 731–740.

Divesh Aggarwal and Ueli Maurer. 2009. Breaking RSA generically is equivalent to factoring. In *Advances in Cryptology – EUROCRYPT'09 (Lecture Notes in Computer Science)*, Antoine Joux (Ed.), Vol. 5479. Springer, 36–53.

Michael Alekhnovich. 2003. More on average case vs approximation complexity. In *44th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 298–307.

Benny Applebaum, Boaz Barak, and David Xiao. 2008. On basing lower-bounds for learning on worst-case assumptions. In *49th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 211–220.

Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. 2009. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology – CRYPTO'09 (Lecture Notes in Computer Science) (LNCS)*, Shai Halevi (Ed.), Vol. 5677. Springer, 595–618.

Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. 2004. Cryptography in NC^0. In *45th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 166–175.

Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. 2006. Computationally private randomizing polynomials and their applications. *Computational Complexity* 15, 2 (2006), 115–162.

Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. 2010. Cryptography by cellular automata or how fast can complexity emerge in nature? In *1st Innovations in Computer Science*, Andrew Chi-Chih Yao (Ed.). Tsinghua University Press, 1–19.

Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. 2011. How to garble arithmetic circuits. In *52nd Annual Symposium on Foundations of Computer Science*, Rafail Ostrovsky (Ed.). IEEE Computer Society Press, 120–129.

Benny Applebaum, Yuval Ishai, Eyal Kushilevitz, and Brent Waters. 2013. Encoding functions with constant online rate or how to compress garbled circuits keys. In *Advances in Cryptology – CRYPTO'13, Part II (Lecture Notes in Computer Science)*, Ran Canetti and Juan A. Garay (Eds.), Vol. 8043. Springer, 166–184.

Theodore P. Baker, John Gill, and Robert Solovay. 1975. Relativizations of the P =? NP question. *SICOMP: SIAM Journal on Computing* 4 (1975), 431–442.

Walter Baur and Volker Strassen. 1983. The complexity of partial derivatives. *Theoretical Computer Science* 22, 3 (1983), 317–330.

Donald Beaver. 1995. Precomputing oblivious transfer. In *Advances in Cryptology – CRYPTO'95 (Lecture Notes in Computer Science)*, Don Coppersmith (Ed.), Vol. 963. Springer, 97–109.

Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. 2012. Foundations of garbled circuits. In *19th Conference on Computer and Communications Security (ACM CCS'12)*, Ting Yu, George Danezis, and Virgil D. Gligor (Eds.). ACM Press, 784–796.

Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. 1988. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*. ACM Press, 1–10.

Rikke Bendlin, Ivan Damgård, Claudio Orlandi, and Sarah Zakarias. 2011. Semi-homomorphic encryption and multiparty computation. In *Advances in Cryptology – EUROCRYPT'11 (Lecture Notes in Computer Science)*, Kenneth G. Paterson (Ed.), Vol. 6632. Springer, 169–188.

Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. 1993. Cryptographic primitives based on hard learning problems. In *Advances in Cryptology – CRYPTO'93 (Lecture Notes in Computer Science)*, Douglas R. Stinson (Ed.), Vol. 773. Springer, 278–291.

Manuel Blum and Silvio Micali. 1982. How to generate cryptographically strong sequences of pseudo random bits. In *23rd Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 112–117.

Andrej Bogdanov and Chin Ho Lee. 2013. Limits of provable security for homomorphic encryption. In *Advances in Cryptology – CRYPTO'13, Part I (Lecture Notes in Computer Science)*, Ran Canetti and Juan A. Garay (Eds.), Vol. 8042. Springer, 111–128.

Zvika Brakerski and Vinod Vaikuntanathan. 2011. Efficient fully homomorphic encryption from (standard) LWE. In *52nd Annual Symposium on Foundations of Computer Science*, Rafail Ostrovsky (Ed.). IEEE Computer Society Press, 97–106.

David Chaum, Claude Crépeau, and Ivan Damgård. 1988. Multiparty unconditionally secure protocols (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*. ACM Press, 11–19.

Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. 2001. Multiparty computation from threshold homomorphic encryption. In *Advances in Cryptology – EUROCRYPT'01 (Lecture Notes in Computer Science)*, Birgit Pfitzmann (Ed.), Vol. 2045. Springer, 280–299.

Ronald Cramer and Serge Fehr. 2002. Optimal black-box secret sharing over arbitrary Abelian groups. In *Advances in Cryptology – CRYPTO'02 (Lecture Notes in Computer Science)*, Moti Yung (Ed.), Vol. 2442. Springer, 272–287.

Ronald Cramer, Serge Fehr, Yuval Ishai, and Eyal Kushilevitz. 2003. Efficient multi-party computation over rings. In *Advances in Cryptology – EUROCRYPT'03 (Lecture Notes in Computer Science)*, Eli Biham (Ed.), Vol. 2656. Springer, 596–613.

Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. 2012. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology – CRYPTO'12 (Lecture Notes in Computer Science)*, Reihaneh Safavi-Naini and Ran Canetti (Eds.), Vol. 7417. Springer, 643–662.

Alexander W. Dent. 2002. Adapting the weaknesses of the random oracle model to the generic group model. In *Advances in Cryptology – ASIACRYPT'02 (Lecture Notes in Computer Science)*, Yuliang Zheng (Ed.), Vol. 2501. Springer, 100–109.

Yvo Desmedt and Yair Frankel. 1991. Shared generation of authenticators and signatures (extended abstract). In *Advances in Cryptology – CRYPTO'91 (Lecture Notes in Computer Science)*, Joan Feigenbaum (Ed.), Vol. 576. Springer, 457–469.

Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. 2008. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing.* 38, 1 (2008), 97–139.

Zeev Dvir, Ariel Gabizon, and Avi Wigderson. 2009. Extractors and rank extractors for polynomial sources. *Computational Complexity* 18, 1 (2009), 1–58.

Zeev Dvir, Dan Gutfreund, Guy N. Rothblum, and Salil P. Vadhan. 2011. On approximating the entropy of polynomial mappings. In *2nd Innovations in Computer Science (ICS'10)*. 460–475.

Richard Ehrenborg and Gian-Carlo Rota. 1993. Apolarity and canonical forms for homogeneous polynomials. *European Journal of Combinatorics* 14, 3 (1993), 157–181.

Taher ElGamal. 1984. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology – CRYPTO'84 (Lecture Notes in Computer Science)*, G. R. Blakley and David Chaum (Eds.), Vol. 196. Springer, 10–18.

Uriel Feige, Joe Kilian, and Moni Naor. 1994. A minimal model for secure computation (extended abstract). In *26th Annual ACM Symposium on Theory of Computing*. ACM Press, 554–563.

Matthew K. Franklin and Stuart Haber. 1993. Joint encryption and message-efficient secure computation. In *Advances in Cryptology – CRYPTO'93 (Lecture Notes in Computer Science)*, Douglas R. Stinson (Ed.), Vol. 773. Springer, 266–277.

Sanjam Garg, Craig Gentry, and Shai Halevi. 2013. Candidate multilinear maps from ideal lattices. In *Advances in Cryptology – EUROCRYPT'13 (Lecture Notes in Computer Science)*, Thomas Johansson and Phong Q. Nguyen (Eds.), Vol. 7881. Springer, 1–17.

Craig Gentry. 2009. Fully homomorphic encryption using ideal lattices. In *41st Annual ACM Symposium on Theory of Computing*, Michael Mitzenmacher (Ed.). ACM Press, 169–178.

Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. 2010. i-Hop homomorphic encryption and rerandomizable Yao circuits. In *Advances in Cryptology – CRYPTO'10 (Lecture Notes in Computer Science)*, Tal Rabin (Ed.), Vol. 6223. Springer, 155–172.

Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. 2008. How to encrypt with the LPN problem. In *35th International Colloquium on Automata, Languages and Programming, Part II (ICALP'08) (Lecture Notes in Computer Science)*, Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz (Eds.), Vol. 5126. Springer, 679–690.

Oded Goldreich. 2001. *Foundations of Cryptography: Basic Tools*. Vol. 1. Cambridge University Press, Cambridge, UK. xix + 372 pages.

Oded Goldreich. 2004. *Foundations of Cryptography: Basic Applications*. Vol. 2. Cambridge University Press, Cambridge, UK.

Oded Goldreich, Shafi Goldwasser, and Silvio Micali. 1986. How to construct random functions. *Journal of the ACM* 33 (1986), 792–807.

Oded Goldreich, Hugo Krawczyk, and Michael Luby. 1988. On the existence of pseudorandom generators. In *Advances in Cryptology – CRYPTO'88 (Lecture Notes in Computer Science)*, Shafi Goldwasser (Ed.), Vol. 403. Springer, 146–162.

Oded Goldreich and Leonid A. Levin. 1989. A hard-core predicate for all one-way functions. In *21st Annual ACM Symposium on Theory of Computing*. ACM Press, 25–32.

Oded Goldreich, Silvio Micali, and Avi Wigderson. 1987. How to play any mental game or a completeness theorem for protocols with honest majority. In *19th Annual ACM Symposium on Theory of Computing*, Alfred Aho (Ed.). ACM Press, 218–229.

Oded Goldreich and Salil P. Vadhan. 1999. Comparing entropies in statistical zero knowledge with applications to the structure of SZK. In *Conference on Computational Complexity (CCC'99)*. 54.

Shafi Goldwasser and Silvio Micali. 1984. Probabilistic encryption. *Journal of Computer and System Sciences* 28, 2 (1984), 270–299.

Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. 1999. A pseudorandom generator from any one-way function. *SIAM Journal on Computing.* 28, 4 (1999), 1364–1396.

Yuval Ishai. 2012. Randomization techniques for secure computation. In *Secure Multi-Party Computation*, Manoj Prabhakaran and Amit Sahai (Eds.). Cryptology and Information Security Series, Vol. 10. IOS Press, Amsterdam, 222–248.

Yuval Ishai and Eyal Kushilevitz. 2000. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 294–304.

Yuval Ishai, Eyal Kushilevitz, Sigurd Meldgaard, Claudio Orlandi, and Anat Paskin-Cherniavsky. 2013. On the power of correlated randomness in secure computation. In *10th Theory of Cryptography Conference (TCC'13) (Lecture Notes in Computer Science)*, Amit Sahai (Ed.), Vol. 7785. Springer, 600–620.

Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. 2005. Sufficient conditions for collision-resistant hashing. In *2nd Theory of Cryptography Conference (TCC'05) (Lecture Notes in Computer Science)*, Joe Kilian (Ed.), Vol. 3378. Springer, 445–456.

Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. 2008. Founding cryptography on oblivious transfer - efficiently. In *Advances in Cryptology – CRYPTO'08 (Lecture Notes in Computer Science)*, David Wagner (Ed.), Vol. 5157. Springer, 572–591.

Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. 2009. Secure arithmetic computation with no honest majority. In *6th Theory of Cryptography Conference (TCC'09) (Lecture Notes in Computer Science)*, Omer Reingold (Ed.), Vol. 5444. Springer, 294–314.

Abhishek Jain, Stephan Krenn, Krzysztof Pietrzak, and Aris Tentes. 2012. Commitments and efficient zero-knowledge proofs from learning parity with noise. In *Advances in Cryptology – ASIACRYPT'12 (Lecture Notes in Computer Science)*, Xiaoyun Wang and Kazue Sako (Eds.), Vol. 7658. Springer, 663–680.

Jonathan Katz and Yehuda Lindell. 2008. *Introduction to Modern Cryptography*. Chapman and Hall/CRC.

Neeraj Kayal. 2009. The complexity of the annihilating polynomial. In *Conference on Computational Complexity (CCC'09)*. IEEE.

Eike Kiltz, Krzysztof Pietrzak, David Cash, Abhishek Jain, and Daniele Venturi. 2011. Efficient authentication from hard learning problems. In *Advances in Cryptology – EUROCRYPT'11 (Lecture Notes in Computer Science)*, Kenneth G. Paterson (Ed.), Vol. 6632. Springer, 7–26.

Ueli M. Maurer. 2005. Abstract models of computation in cryptography (invited paper). In *10th IMA International Conference on Cryptography and Coding (Lecture Notes in Computer Science)*, Nigel P. Smart (Ed.), Vol. 3796. Springer, 1–12.

Ueli M. Maurer and Stefan Wolf. 1998. Lower bounds on generic algorithms in groups. In *Advances in Cryptology – EUROCRYPT'98 (Lecture Notes in Computer Science)*, Kaisa Nyberg (Ed.), Vol. 1403. Springer, 72–84.

Moni Naor. 1991. Bit commitment using pseudorandomness. *Journal of Cryptology* 4, 2 (1991), 151–158.

Moni Naor and Benny Pinkas. 1999. Oblivious transfer and polynomial evaluation. In *31st Annual ACM Symposium on Theory of Computing*. ACM Press, 245–254.

Krzysztof Pietrzak. 2012. Cryptography from learning parity with noise. In *Theory and Practice of Computer Science - Proceedings of the 38th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM'12)*. 99–114.

Alexander A. Razborov and Steven Rudich. 1994. Natural proofs. In *26th Annual ACM Symposium on Theory of Computing*. ACM Press, 204–213.

Oded Regev. 2005. On lattices, learning with errors, random linear codes, and cryptography. In *37th Annual ACM Symposium on Theory of Computing*, Harold N. Gabow and Ronald Fagin (Eds.). ACM Press, 84–93.

Jacob T. Schwartz. 1980. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM (JACM)* 27, 4 (1980), 701–717.

Adi Shamir. 1979. How to share a secret. *Communications of the Association for Computing Machinery* 22, 11 (Nov. 1979), 612–613.

Victor Shoup. 1997. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology – EUROCRYPT'97 (Lecture Notes in Computer Science)*, Walter Fumy (Ed.), Vol. 1233. Springer, 256–266.

Amir Shpilka and Amir Yehudayoff. 2010. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science* 5, 3–4 (2010), 207–388.

Atri Rudra Venkatesan Guruswami and Madhu Sudan. 2014. *Essential Coding Theory (Unpublished Book)*. http://www.cse.buffalo.edu/~atri/courses/coding-theory/book/.

Andrew Chi-Chih Yao. 1982. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 80–91.

Andrew Chi-Chih Yao. 1986. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 162–167.

Richard Zippel. 1979. Probabilistic algorithms for sparse polynomials. In *Symbolic and Algebraic Computation (EUROSAM'79)*. Springer, Marseille, France, 216–226.

Received April 2015; revised December 2016; accepted January 2017