Arithmetic Cryptography

[Extended Abstract] *

Benny Applebaum
Tel-Aviv University
bennyap@post.tau.ac.il

Jonathan Avron
Tel-Aviv University
avron@mail.tau.ac.il

Christina Brzuska†
Microsoft Cambridge
christina.brzuska@gmail.com

ABSTRACT

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 which 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 reasonable 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, homomorphic encryption, and secure computation with low online complexity cannot be achieved in this model. Our results reveal a qualitative difference between the standard model and the arithmetic model, and explain, in retrospect, some of the limitations of previous constructions.

Categories and Subject Descriptors

F.1.2 [**Modes of Computation**]: Interactive and reactive computation

General Terms

Theory of Computation

*A full version of this paper is available at `http://www.eng.tau.ac.il/ bennyap/publications.html`. This research was supported by ISF grant 1155/11, Israel Ministry of Science and Technology (grant 3-9094), GIF grant 1152/2011, and by the Check Point Institute for Information Security.

†Work done while being a postdoctoral fellow at Tel Aviv University.

ITCS'15, January 11–13, 2015, Rehovot, Israel.

ACM 978-1-4503-3333-7/15/01?$15.00.
http://dx.doi.org/10.1145/2688073.2688114.

Keywords

Cryptography; computational complexity; arithmetic circuits

1. INTRODUCTION

This paper studies the possibility of solving cryptographic problems in a way which 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) [27]. 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.

One way to solve the problem is to translate x and f to binary form. Unfortunately, this solution suffers from several limitation. First, such a translation is typically expensive as it introduces a large overhead (typically much larger than $\log |\mathbb{F}|$).[1] Secondly, 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 an 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 [26]).

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 which can be later instantiated with any concrete domain. In this paper we explore the feasibility of such schemes. More generally, we study the following natural question:

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

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

We formalize the above 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 which, 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 an access to the bit representation of the inputs or even to the size of $\mathbb{F}$. We 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*.

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

EXAMPLE 1.1 (ONE-TIME ENCRYPTION). *We illustrate the model by defining 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 *which satisfy the perfect correctness condition:*

$$\Pr_{k \xleftarrow{R} \mathsf{KGen}(1^n)} [\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = m] = 1,$$

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 $k \xleftarrow{R} \mathsf{KGen}(1^n)$ and $b \xleftarrow{R} \{0,1\}$; (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 $k \xleftarrow{R} \mathbb{F}^n$, 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 the above 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 paper is to find out which cryptographic primitives admit arithmetic constructions. We begin by observing that, similarly to the case of one-time pad, typical information-theoretic constructions naturally arithmetize. Notable examples include various secret sharing schemes [47, 21, 17], and classical information-theoretic secure multiparty protocols [10, 15]. (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 scheme, 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 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 $y \in \mathsf{Span}(M)$ and replacing a random εm-subset of y's coordinates with random field elements.[3] This Random-Linear-Code assumption, which is denoted by $\mathsf{RLC}_{\mathbb{F}}(n, m, \varepsilon)$,

[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).

[3]This is contrasted with the more standard Learning-With-Errors (LWE) assumption [46] in which *each* coordinate of y is perturbed with some "small" element from the ring $\mathbb{Z}_p$, e.g., 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 which grows with p, and so LWE constructions do not seem to arithmetize. See Section 1.3 for further discussion.

was previously considered in [39]. If $\mathbb{F}$ is instantiated with the binary field, we get the standard *Learning Parity with Noise* (LPN) assumption [30, 12]. Indeed, some of the primitives in the above theorem can be constructed by extending various LPN-based schemes from the literature.

Theorem 1.1 shows that the arithmetic model is rich enough to allow highly non-trivial 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 which can be achieved in the standard model but cannot be implemented arithmetically. This include garbled circuits, secure computation protocols with "low" online communication, and multiplicative homomorphic encryption schemes. Details follow.

Garbled circuits.

Yao's *garbled circuit* (GC) construction [50] 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* which recovers $C(x)$ from $(\hat{C}, K_x)$ and an efficient *simulator* which, given $C(x)$, samples from a distribution which 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 [13, 49], or equivalently a one-way function [34].

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 in the security parameter (i.e., can be taken to be n^ε for an arbitrary $\varepsilon > 0$). 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. [9].)

The question of garbling arithmetic circuits has been open for a long time, and only recently some partial progress has been made [5]. Still, so far there has been no fully arithmetic construction in which both the encoder and the decoder make a black-box use of $\mathbb{F}$. We show that this is inherently impossible answering an open problem from [35].

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]

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

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.

Secure computation with low online complexity.

Generalizing the above result, we prove a non-trivial lower-bound on the online communication complexity of semi-honest secure computation protocols. Roughly speaking, we allow the parties to exchange all the messages which 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 [8, 38, 11, 19, 37]. In the standard Boolean setting, there are protocols which achieve highly efficient online communication complexity. For example, for efficient deterministic two-party functionalities $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ which deliver the output to one of the parties (hereafter referred to as simple functionalities), one can obtain semi-honest 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 [6]. 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 standard one-way functions exist. Then, for every constant $c > 0$ there exists a simple arithmetic two-party functionality $f : \mathbb{F}^n \times \mathbb{F}^n \to \mathbb{F}^{n^c}$ which cannot be securely computed by an arithmetic semi-honest protocol with online communication smaller than $\Omega(n^c)$ field elements.*

The theorem generalizes to the multiparty setting including the case of honest majority.

Multiplicative homomorphic encryption.

A 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 transformation T such that, for every messages $x, a \in \mathbb{F}$ and almost all public keys pk, the distributions

$$(c, c') \qquad \text{and} \qquad (c, c''), \tag{1}$$

where $c = \mathsf{Enc}_{\mathsf{pk}}(x), c' = T(\mathsf{pk}, c, a))$ and $c'' = \mathsf{Enc}_{\mathsf{pk}}(a \cdot x)$, are identical.[5] Two well known examples for such schemes (over different fields) are Goldwasser-Micali cryptosystem [33] and ElGamal cryptosystem [23].

We show that multiplicative homomorphic encryption cannot be implemented arithmetically. Unlike the previous theorems, our proof holds only in a *strict* arithmetic model where the honest algorithms are not allowed to use zero-testing gates and division gates.

Informal Theorem 1.4. *There are no perfectly-correct multiplicative homomorphic encryption schemes in the strict*

[5]For technical reasons, we further require the encryption to be *regular*, meaning that the distribution of a random encryption $\mathsf{Enc}_{\mathsf{pk}}(x)$ of a random field element $x \xleftarrow{R} \mathbb{F}$ should be (close to) uniform over the image of $\mathsf{Enc}_{\mathsf{pk}}$.

arithmetic setting. Furthermore, this holds even if the decryption algorithm is non-arithmetic or even inefficient.

The case of inefficient decryption algorithm corresponds to non-interactive perfectly binding *commitments* with multiplicative homomorphism. Interestingly, the commitments constructed in Theorem 1.1 (which are strictly arithmetic, non-interactive, perfectly binding, and regular) enjoy *weak multiplicative homomorphism*. Namely, only the marginals c' and c'', defined in (1), are identically distributed. So the main issue seems to be *strong* homomorphism, which cannot be achieved arithmetically, but can be easily achieved (for scalar multiplication) in the Boolean setting.

1.3 Discussion

Taken together, our positive and negative results suggest that the arithmetic model is highly non-trivial yet significantly weaker from 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, [5] 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 to garble circuits over the ring $\mathbb{Z}_p$ in a "semi-arithmetic" model, in which the encoder can treat the inputs as integers and the decoder is non-arithmetic. 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 explains the high communication complexity of arithmetic MPC protocols such as the ones from [10, 15, 18, 39].

Moreover, we believe that our results have interesting implications regarding the standard *Boolean* model. Inspired by computational complexity theory [7, 45, 1], one can view our negative results as some form of a barrier.

> **The Arithmetization Barrier:** If your construction "arithmetize" then it faces the lower-bounds.

LPN/RLC vs. LWE.

As an example, it seems that constructions which 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 commitment with (strong) multiplicative homomorphism. Since such schemes can be easily constructed under Regev's Learning-With-Errors (LWE) assumption [46], this exposes a qualitative difference between the two assumptions. Indeed, this gap between strong LWE-type homomorphism (as in Eq. 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 [14]), 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 strong homomorphism is not just a limitation of a *concrete* construction, but it is, in fact, inherent to *all* arithmetic constructions. Quoting Pietrzak [44] one may wonder: "... is there a fundamental reason why the more general LWE problem allows for such objects, but LPN does not?" A simple answer would be: "LPN arithmetize 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, in the area of IT-secure primitives, proving lower-bounds (even non-constructively) is notoriously hard.[6] The arithmetic model restricts the honest parties, and as a result makes lower-bounds much 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 non-arithmetic route.

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 [47], the classical information-theoretic secure multiparty protocols of [10, 15] and the randomized encodings of [36]. Extensions of these results to generic black-box *rings* were given in [21, 17, 18].

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 have *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, Prabhakaran and Sahai [39] (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 [43] yields semi-honest secure two-party protocols in the IPS model. Security against

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

[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.

malicious adversaries (as well as generalization to general rings and efficiency improvements) were given by [39]. Both works rely on the existence of a Boolean Oblivious Transfer primitive.

Arithmetic reductions.

Another line of works 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 [25] for the semi-honest model, and were extended by [16] to the malicious model (assuming that the underlying encryption is equipped with special-purpose zero-knowledge protocols). Similarly, the results of [5] 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 [48, 42, 41, 2]. 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 non-arithmetic and has the power to specify the field and its representation. These two models also serve very different purposes: The GGM allows to prove unconditional hardness results against "generic attacks", while our model allows 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 [20].

2. TECHNIQUES: NEGATIVE RESULTS

In 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. In a sense, our lower-bounds show that *algorithms for arithmetic circuits can be used to attack arithmetic constructions.* Below we give an outline of the proofs of the main negative results.

For ease of presentation, we sketch (in Section 2.1) a version of Theorems 1.2 and 1.3 in the Private Simultaneous Messages (PSM) model of [24], which is conceptually simpler than garbled circuits and general secure computation protocols. Section 2.2 contains an overview of the proof of Theorem 1.4.

2.1 Lower Bounds in the PSM model

The PSM model.

Consider two parties Alice and Bob that 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 $|x| \cdot k$ [24], or even $|x| + k$ [6]. 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 non-trivial information about Bob's input. In patricular, Carol will output a non-zero vector which is orthogonal to y_1. (This clearly violates privacy as it allows Carol to exclude $1/|\mathbb{F}|$ 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 (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 x. Similarly, the Jacobian matrix $\mathcal{J}C_{\mathbf{b}}(a) : \mathbb{F}^{n-1} \to \mathbb{F}^{n \times n-1}$ is a matrix of multivariate polynomials whose (i, j)-th entry is the partial derivative of the i-th output of $C_{\mathbf{b}}(a) : \mathbb{F}^{n-1} \to \mathbb{F}^n$ with respect to the j-th 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 neither $A_{\mathbf{r}}(x)$ nor

[8] We use bold fonts for fixed value, and standard fonts for non-fixed 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 $2s = \text{poly}(n)$.

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 (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 non-zero vector which 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, (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 which is applied to the expression $Q(x)$. Note that $Q(x)$ is a polynomial of degree d which 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. The above 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 an 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 which is applied to a non-zero polynomial Q which vanishes over the image of $A_{\mathbf{r}}$. In this case, the above 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 which 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 above 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 standard (Boolean) pseudorandom generator (PRG). Roughly speaking, we embed a binary PRG in the function f such that a low communication protocol allows to break the pseudroandomness 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.

2.2 Impossibility of Multiplicative Homomorphic Encryption

To prove Theorem 1.4 we show that in order to attack multiplicative homomorphic encryption, it suffices to estimate the entropy of some probability distribution which is represented by a given arithmetic circuit. The idea is simple: given a public-key pk and an encryption $C = \mathsf{Enc}_{\mathsf{pk}}(y)$ of an unknown plaintext $y \in \{0,1\}$, we use the multiplicative homomorphism to construct the circuit $f_{C,\mathsf{pk}}(x)$ which maps a plaintext $x \in \mathbb{F}$ into a fresh encryption of $x \cdot y$. Consider the probability distribution of $f_{C,\mathsf{pk}}(x)$ induced by a uniform choice of $x \xleftarrow{R} \mathbb{F}$ 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 $f_{C,\mathsf{pk}}(x)$ is simply a fresh encryption of the zero element. In contrast, if C is an encryption of 1 (or any non-zero element) then $f_{C,\mathsf{pk}}(x)$ is a fresh encryption of a random field element. Intuitively, the latter distribution should have larger entropy than the first one. At this point we employ the algorithm of Dvir et al. [22] which, given an arithmetic circuit f, estimates the *min-entropy* of the distribution sampled by f.[10]

3. TECHNIQUES: POSITIVE RESULTS

Our positive results (Theorem 1.1) are based on three different approaches – outlined below.

3.1 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 which allows 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 be therefore viewed as a bridge between the binary world to 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 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' which 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" de-

[10]The min-entropy of a probability distribution D measures (in logarithmic scale) the weight of the heaviest element in D.

[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 $\mathbb{F}$.

cryption 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 $\mathsf{RLC}_{\mathbb{F}}(n, m, \varepsilon)$. To encrypt a message x, sample a random generating matrix $A \xleftarrow{R} \mathbb{F}^{m \times n}$ together with a random ε-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 ε, 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 [39]. Recall that a (binary) one-out-of-two oblivious transfer ($\binom{2}{1}$-OT) is a two-party functionality which 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 [18]).[12] This gives a semi-honest OLE.

At this point, we can use the semi-honest OLE together with an arithmetic variant of the classical GMW protocol [32] to obtain an arithmetic secure computation protocol for general arithmetic functions in the semi-honest model. This protocol can be transformed to the malicious setting using the IPS compiler [38]. 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 [10] or [18]), and a maliciously-secure $\binom{2}{1}$-AOT (which we already constructed).

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

3.2 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 [28], commitments [4, 40], and even public-key encryption [3] 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 $\mathsf{RLC}(n, m, \varepsilon)$ for constant-rate code $m = O(n)$ and sub-constant noise rate $\varepsilon = O(1/\sqrt{n})$. In the case of CPA-secure symmetric encryption, the direct approach requires hardness for *any* polynomial $m = m(n)$ and constant noise ε. In contrast, for both primitives, the ABE-based approach requires only hardness for constant rate codes $m = O(n)$ and constant noise rate ε. While all three assumptions are consistent with our knowledge, the third assumption is formally weaker than (i.e., implied by) the first two.

Arithmetizing Cryptographic Transformations.

Another way to construct arithmetic primitives is to start with some direct 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 [32] of semi-honest secure computation from oblivious transfer (OT) naturally extends to the arithmetic setting [39]. 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 to zero or x_i equals to 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 [29] of pseudorandom functions (PRFs) from pseudorandom generators (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 i-th step is chosen based on the i-th 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 [31], 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.

Acknowledgment

We thank Yuval Ishai for useful discussions.

4. REFERENCES

[1] S. Aaronson and A. Wigderson. Algebrization: a new barrier in complexity theory. In R. E. Ladner and C. Dwork, editors, *40th ACM STOC*, pages 731–740. ACM Press, May 2008.

[2] D. Aggarwal and U. Maurer. Breaking RSA generically is equivalent to factoring. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 36–53. Springer, Apr. 2009.

[3] M. Alekhnovich. More on average case vs approximation complexity. In *44th FOCS*, pages 298–307. IEEE Computer Society Press, Oct. 2003.

[4] B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography by cellular automata or how fast can complexity emerge in nature? In A. C.-C. Yao, editor, *ICS 2010*, pages 1–19. Tsinghua University Press, Jan. 2010.

[5] B. Applebaum, Y. Ishai, and E. Kushilevitz. How to garble arithmetic circuits. In R. Ostrovsky, editor, *52nd FOCS*, pages 120–129. IEEE Computer Society Press, Oct. 2011.

[6] B. Applebaum, Y. Ishai, E. Kushilevitz, and B. Waters. Encoding functions with constant online rate or how to compress garbled circuits keys. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 166–184. Springer, Aug. 2013.

[7] Baker, Gill, and Solovay. Relativizations of the P =? NP question. *SICOMP: SIAM Journal on Computing*, 4, 1975.

[8] D. Beaver. Precomputing oblivious transfer. In D. Coppersmith, editor, *CRYPTO'95*, volume 963 of *LNCS*, pages 97–109. Springer, Aug. 1995.

[9] M. Bellare, V. T. Hoang, and P. Rogaway. Foundations of garbled circuits. In T. Yu, G. Danezis, and V. D. Gligor, editors, *ACM CCS 12*, pages 784–796. ACM Press, Oct. 2012.

[10] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

[11] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias. Semi-homomorphic encryption and multiparty computation. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 169–188. Springer, May 2011.

[12] A. Blum, M. L. Furst, M. J. Kearns, and R. J. Lipton. Cryptographic primitives based on hard learning problems. In D. R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 278–291. Springer, Aug. 1993.

[13] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo random bits. In *23rd FOCS*, pages 112–117. IEEE Computer Society Press, Nov. 1982.

[14] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In R. Ostrovsky, editor, *52nd FOCS*, pages 97–106. IEEE Computer Society Press, Oct. 2011.

[15] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *20th ACM STOC*, pages 11–19. ACM Press, May 1988.

[16] R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty computation from threshold homomorphic encryption. In B. Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 280–299. Springer, May 2001.

[17] R. Cramer and S. Fehr. Optimal black-box secret sharing over arbitrary Abelian groups. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 272–287. Springer, Aug. 2002.

[18] R. Cramer, S. Fehr, Y. Ishai, and E. Kushilevitz. Efficient multi-party computation over rings. In E. Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 596–613. Springer, May 2003.

[19] I. Damgård, V. Pastro, N. P. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 643–662. Springer, Aug. 2012.

[20] A. W. Dent. Adapting the weaknesses of the random oracle model to the generic group model. In Y. Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 100–109. Springer, Dec. 2002.

[21] Y. Desmedt and Y. Frankel. Shared generation of authenticators and signatures (extended abstract). In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 457–469. Springer, Aug. 1991.

[22] Z. Dvir, A. Gabizon, and A. Wigderson. Extractors and rank extractors for polynomial sources. In *48th FOCS*, pages 52–62. IEEE Computer Society Press, Oct. 2007.

[23] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and D. Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 10–18. Springer, Aug. 1984.

[24] U. Feige, J. Kilian, and M. Naor. A minimal model for secure computation (extended abstract). In *26th ACM STOC*, pages 554–563. ACM Press, May 1994.

[25] M. K. Franklin and S. Haber. Joint encryption and message-efficient secure computation. In D. R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 266–277. Springer, Aug. 1993.

[26] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 1–17. Springer, May 2013.

[27] C. Gentry. Fully homomorphic encryption using ideal lattices. In M. Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

[28] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. How to encrypt with the LPN problem. In L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, editors, *ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 679–690. Springer, July 2008.

[29] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33:792–807, 1986.

[30] O. Goldreich, H. Krawczyk, and M. Luby. On the existence of pseudorandom generators. In S. Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 146–162. Springer, Aug. 1988.

[31] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989.

[32] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In A. Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[33] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[34] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[35] Y. Ishai. Randomization techniques for secure computation. In M. Prabhakaran and A. Sahai, editors, *Secure Multi-Party Computation*, volume 10 of *Cryptology and Information Security Series*, pages 222–248. IOS press, Amsterdam, 2012.

[36] Y. Ishai and E. Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st FOCS*, pages 294–304. IEEE Computer Society Press, Nov. 2000.

[37] Y. Ishai, E. Kushilevitz, S. Meldgaard, C. Orlandi, and A. Paskin-Cherniavsky. On the power of correlated randomness in secure computation. In A. Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 600–620. Springer, Mar. 2013.

[38] Y. Ishai, M. Prabhakaran, and A. Sahai. Founding cryptography on oblivious transfer - efficiently. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, Aug. 2008.

[39] Y. Ishai, M. Prabhakaran, and A. Sahai. Secure arithmetic computation with no honest majority. In O. Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 294–314. Springer, Mar. 2009.

[40] E. Kiltz, K. Pietrzak, D. Cash, A. Jain, and D. Venturi. Efficient authentication from hard learning problems. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 7–26. Springer, May 2011.

[41] U. M. Maurer. Abstract models of computation in cryptography (invited paper). In N. P. Smart, editor, *10th IMA International Conference on Cryptography and Coding*, volume 3796 of *LNCS*, pages 1–12. Springer, Dec. 2005.

[42] U. M. Maurer and S. Wolf. Lower bounds on generic algorithms in groups. In K. Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 72–84. Springer, May / June 1998.

[43] M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *31st ACM STOC*, pages 245–254. ACM Press, May 1999.

[44] K. Pietrzak. Cryptography from learning parity with noise. In *SOFSEM 2012: Theory and Practice of Computer Science - 38th Conference on Current Trends in Theory and Practice of Computer Science, Špindleruv Mlýn, Czech Republic, January 21-27, 2012. Proceedings*, pages 99–114, 2012.

[45] A. A. Razborov and S. Rudich. Natural proofs. In *26th ACM STOC*, pages 204–213. ACM Press, May 1994.

[46] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In H. N. Gabow and R. Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[47] A. Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, Nov. 1979.

[48] V. Shoup. Lower bounds for discrete logarithms and related problems. In W. Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer, May 1997.

[49] A. C.-C. Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd FOCS*, pages 80–91. IEEE Computer Society Press, Nov. 1982.

[50] A. C.-C. Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, Oct. 1986.