Secure Arithmetic Computation with No Honest Majority*

Yuval Ishai[1,**], Manoj Prabhakaran[2,***], and Amit Sahai[3,†]

[1] Technion, Israel and University of California, Los Angeles
yuvali@cs.technion.il
[2] University of Illinois, Urbana-Champaign
mmp@cs.uiuc.edu
[3] University of California, Los Angeles
sahai@cs.ucla.edu

Abstract. We study the complexity of securely evaluating arithmetic circuits over finite rings. This question is motivated by natural secure computation tasks. Focusing mainly on the case of *two-party* protocols with security against *malicious* parties, our main goals are to: (1) only make black-box calls to the ring operations and standard cryptographic primitives, and (2) minimize the number of such black-box calls as well as the communication overhead.

We present several solutions which differ in their efficiency, generality, and underlying intractability assumptions. These include:

- An *unconditionally secure* protocol in the OT-hybrid model which makes a black-box use of an arbitrary ring R, but where the number of ring operations grows linearly with (an upper bound on) $\log |R|$.
- Computationally secure protocols in the OT-hybrid model which make a black-box use of an underlying ring, and in which the number of ring operations does not grow with the ring size. The protocols rely on variants of previous intractability assumptions related to linear codes. In the most efficient instance of these protocols, applied to a suitable class of fields, the (amortized) communication cost is a constant number of field elements per multiplication gate and the computational cost is dominated by $O(\log k)$ field operations per gate, where k is a security parameter. These results extend a previous approach of Naor and Pinkas for secure polynomial evaluation (*SIAM J. Comput.*, 2006).
- A protocol for the rings $\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$ which only makes a black-box use of a homomorphic encryption scheme. When m is prime, the

* Extended Abstract. Please see full version at Cryptology ePrint Archive: Report 2008/465.

** Supported in part by ISF grant 1310/06, BSF grant 2004361, and NSF grants 0205594, 0430254, 0456717, 0627781, 0716389.

*** Supported in part by NSF grants CNS 07-16626 and CNS 07-47027.

† Research supported in part from NSF grants 0627781, 0716389, 0456717, and 0205594, BSF grant 2004361, a subgrant from SRI as part of the Army Cyber-TA program, an equipment grant from Intel, an Alfred P. Sloan Foundation Fellowship, and an Okawa Foundation Research Grant.

O. Reingold (Ed.): TCC 2009, LNCS 5444, pp. 294–314, 2009.

(amortized) number of calls to the encryption scheme for each gate of the circuit is constant.

All of our protocols are in fact *UC-secure* in the OT-hybrid model and can be generalized to *multiparty* computation with an arbitrary number of malicious parties.

1 Introduction

This paper studies the complexity of secure multiparty computation (MPC) tasks which involve *arithmetic* computations. Following the general feasibility results from the 1980s [60,34,4,13], much research in this area shifted to efficiency questions, with a major focus on the efficiency of securely distributing natural computational tasks that arise in the "real world". In many of these cases, some inputs, outputs, or intermediate values in the computation are integers, finite-precision reals, matrices, or elements of a big finite ring, and the computation involves arithmetic operations in this ring. To name just a few examples from the MPC literature, such arithmetic computations are useful in the contexts of distributed generation of cryptographic keys [8,28,56,32,2], privacy-preserving data-mining and statistics [48,11], comparing and matching data [50,31,38], auctions and mechanism design [51,21,59,7], and distributed linear algebra computations [15,52,47,20,49].

This motivates the following question:

> What is the complexity of securely evaluating a given arithmetic circuit C over a given finite ring R?

Before surveying the state of the art, some clarifications are in place.

Arithmetic circuits. An arithmetic circuit over a ring is defined similarly to a standard boolean circuit, except that the inputs and outputs are ring elements rather than bits and gates are labeled by the ring operations add, subtract, and multiply. (Here and in the following, by "ring" we will refer to a finite ring by default.) In the current context of distributed computations, the inputs and outputs of the circuit are annotated with the parties to which they belong. Thus, the circuit C together with the ring R naturally define a multi-party arithmetic functionality C^R. Note that arithmetic computations over the integers or finite-precision reals can be embedded into a sufficiently large finite ring or field, provided that there is an a-priori upper bound on the bit-length of the output. See Section 1.2 for further discussion of the usefulness of arithmetic circuits and some extensions of this basic model to which our results apply.

Secure computation model. The main focus of this paper is on secure *two-party* computation or, more generally, MPC with an arbitrary number of malicious parties. (In this setting it is generally impossible to guarantee output delivery or even fairness, and one has to settle for allowing the adversary to abort the protocol after learning the output.) Our protocols are described in the "OT-hybrid model," namely in a model that allows parties to invoke an

ideal oblivious transfer (OT) oracle [57,27,33]. This has several advantages in generality and efficiency, see [44] and Section 1.2 for discussion.

Ruling out the obvious. An obvious approach for securely realizing an arithmetic computation C^R is by first designing an equivalent *boolean* circuit C' which computes the same function on a binary representation of the inputs, and then using standard MPC protocols for realizing C'. The main disadvantage of such an approach is that it typically becomes very inefficient when R is large. A clean way for ruling out such an approach, which is of independent theoretical interest, is by restricting protocols to only make a *black-box* access to the ring R. That is, Π securely realizes C if Π^R securely realizes C^R for every finite ring R and *every representation of elements in* R. The black-box access to R enables Π to perform ring operations and sample random ring elements, but the correspondence between ring elements and their identifiers (or even the exact size of the ring) will be unknown to the protocol. This automatically ensures that the overhead of Π (compared to an insecure implementation) does not grow with the computational complexity of ring operations. When considering the special case of fields, we allow by default the protocol Π to access an inversion oracle. Most of our protocols will make black-box access to a ring, although we will also consider some protocols outside of this model based on homomorphic encryption (see below).

1.1 Previous Work

In the setting of MPC *with honest majority*, most protocols from the literature can make a black-box use of an arbitrary *field*. An extension to arbitrary black-box rings was given in [19], building on previous black-box secret sharing techniques of [26,18] and previous MPC techniques of [4,16].

In the case of secure two-party computation and MPC with no honest majority, most protocols from the literature apply to boolean circuits. Below we survey some previous approaches from the literature that apply to secure arithmetic computation with no honest majority.

In the semi-honest model, it is easy to employ any homomorphic encryption scheme with plaintext group $\mathbb{Z}_m$ for performing arithmetic MPC over $\mathbb{Z}_m$. (See, e.g., [1,11].) An alternative approach, which relies on oblivious transfer and uses the standard binary representation of elements in $\mathbb{Z}_m$, was employed in [32]. Both of the above protocols make a black-box use of the underlying cryptographic primitives but do not make a black-box use of the underlying ring. Applying the general compilers of [34,12] to these protocols in order to obtain security in the malicious model would result in inefficient protocols which make a non-black-box use of the underlying cryptographic primitives (let alone the ring).

In the *malicious model*, protocols for secure arithmetic computation based on *threshold* homomorphic encryption were given in [17,24][1] (extending a similar protocol for the semi-honest model from [29]). These protocols provide the

[1] While [17,24] refer to the case of robust MPC in the presence of an honest majority, these protocols can be easily modified to apply to the case of MPC with no honest majority.

most practical general solutions for secure arithmetic two-party computation we are aware of, requiring a constant number of modular exponentiations for each arithmetic gate. On the down side, these protocols require a nontrivial setup of keys which is expensive to distribute. Moreover, they rely on special-purpose zero-knowledge proofs and specific number-theoretic assumptions and thus do not make a black-box use of the underlying cryptographic primitives, let alone a black-box use of the ring.

The only previous approach which makes a black-box use of an underlying ring (as well as a black-box use of OT) was suggested by Naor and Pinkas [50] in the context of secure polynomial evaluation. Their protocol can make a black-box use of any *field*, and its security is related to the conjectured intractability of decoding Reed-Solomon codes with a sufficiently high level of random noise. The protocol from [50] can be easily used to obtain general secure protocols for arithmetic circuits in the *semi-honest* model. However, extending it to allow full simulation-based security in the malicious model (while still making only a black-box use of the underlying field) is not straightforward. (Even in the special case of secure polynomial evaluation, an extension to the malicious model suggested in [50] only considers *privacy* rather than full simulation-based security.)

Finally, we note that Yao's garbled circuit technique [60], the main known technique for constant-round secure computation of general functionalities, does not have a known arithmetic analogue. Thus, in all general-purpose protocols for secure arithmetic computation (including the ones presented in this work) the round complexity must grow with the multiplicative depth of C – the maximal number of multiplication gates on a path from an input to an output.

1.2 Our Contribution

We study the complexity of general secure arithmetic computation over finite rings in the presence of an arbitrary number of malicious parties. We are motivated by the following two related goals.

- *Black-box feasibility*: only make a black-box use of an underlying ring R or field F and standard cryptographic primitives;
- *Efficiency*: minimize the number of such black-box calls, as well as the communication overhead.

For simplicity, we do not attempt to optimize the dependence of the complexity on the number of parties, and restrict the following discussion to the two-party case.

We present several solutions which differ in their efficiency, generality, and underlying intractability assumptions. All these constructions use the general framework from [44]: one can obtain 2-party UC-secure protocols in the OT-hybrid model, by combining an "outer MPC protocol" secure against active adversaries in the *honest majority setting*, with an "inner two-party protocol" for simple functionalities that need only be secure against *passive* adversaries. The main technical contribution of this work is in designing inner protocols, that can then be combined with appropriate variants of outer protocols from the

literature, to obtain secure protocols with desired properties. Below we describe the main protocols we obtain in this way, along with their efficiency and security features.

An unconditionally secure protocol. We present an *unconditionally secure* protocol in the OT-hybrid model which makes a *black-box* use of an *arbitrary* finite ring R, but where the number of ring operations and the number of ring elements being communicated grow linearly with (an upper bound on) $\log |R|$. (We assume for simplicity that an upper bound on $\log |R|$ is given by the ring oracle, though such an upper bound can always be inferred from the length of the strings representing ring elements.) More concretely, the number of ring operations for each gate of C is $poly(k) \cdot \log |R|$, where k is a statistical security parameter. This gives a two-party analogue for the MPC protocol over black-box rings from [19], which requires an honest majority (but does not require the number of ring operations to grow with $\log |R|$).

Protocols based on noisy linear encodings. Motivated by the goal of reducing the overhead of the previous protocol, we present a general approach for deriving secure arithmetic computation protocols over a ring R from linear codes over R. The (computational) security of the protocols relies on intractability assumptions related to the hardness of decoding in the presence of random noise. These protocols generalize and extend in several ways the previous approach of Naor and Pinkas for secure polynomial evaluation [50]. More concretely, we make three main observations: (1) Using [44], secure evaluation of *degree-1* polynomials in the *semi-honest* model can be used in a black-box way for general secure arithmetic computation in the malicious model; (2) In the case of degree-1 polynomials, the approach of [50] can be generalized to rely on *arbitrary* linear codes for which the relevant intractability assumption holds; (3) When using Reed-Solomon codes as in [50], it is possible to significantly improve the efficiency by batching together many instances of secure polynomial evaluation.

Using this approach, we obtain the following types of protocols in the OT-hybrid model.

– A protocol which makes a black-box use of an arbitrary *field* F, in which the number of field operations (and field elements being communicated) does not grow with the field size. More concretely, the number of field operations for each gate of C is bounded by a fixed polynomial in the security parameter k, independently of $|F|$. The underlying assumption is related to the conjectured intractability of decoding a random linear code over F. Our assumption is implied by the assumption that a noisy codeword in a random linear code over F is pseudorandom.

– A variant of the previous protocol which makes a black-box use of an arbitrary *ring* R, and in particular does not rely on inversion. This variant is based on families of linear codes over rings in which decoding in the presence of erasures can be done efficiently, and for which decoding in the presence of (a suitable distribution of) random noise seems intractable.

– The most efficient protocol we present relies on the intractability of decoding Reed-Solomon codes with a (small) constant rate in the presence of a (large)

constant fraction of noise.[2] The amortized communication cost is a constant number of field elements per multiplication gate. (Here and in the following, when we refer to "amortized" complexity we ignore an additive term that may depend on the security parameter and the circuit depth, but not on the circuit size. In most natural instances of large circuits this additive term does not form an efficiency bottleneck.)
A careful implementation yields protocols whose amortized *computational* cost is $O(\log k)$ field operations per gate, where k is a security parameter, assuming that the field size is super-polynomial in k. In contrast, protocols which are based on homomorphic encryption schemes (such as [17] or the ones obtained in this work) apply modular exponentiations, which require $\Omega(k + \log |F|)$ ring multiplications per gate, in a ciphertext ring which is larger than F. This is the case even in the semi-honest model.

Protocols making a black-box use of homomorphic encryption. We also consider protocols for the specific rings $\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$ (thus leaving behind the black-box ring model), but which make black-box use of any homomorphic encryption scheme with plaintext group $\mathbb{Z}_m$. Alternatively, the protocol can make a black-box use of homomorphic encryption schemes in which the plaintext group is determined by the key generation algorithm, such as those of Paillier [53] or Damgård-Jurik [23]. In both variants of the protocol, the (amortized) number of communicated ciphertexts and calls to the encryption scheme for each gate of C is constant, assuming that m is prime. This efficiency feature is comparable to the protocols from [17,24] discussed in Section 1.1 above. Our protocols have the advantages of using a more general primitive and only making a *black-box* use of this primitive (rather than relying on special-purpose zero-knowledge protocols). Furthermore, the additive term which we ignore in the above "amortized" complexity measure seems to be considerably smaller than the cost of distributing the setup of the threshold cryptosystem required by [17].

Both variants of the protocol can be naturally extended to the case of matrix rings $\mathbb{Z}_m^{n \times n}$, increasing the communication complexity by a factor of n^2. (Note that emulating matrix operations via basic arithmetic operations over $\mathbb{Z}_m$ would result in a bigger overhead, corresponding to the complexity of matrix multiplication.) Building on the techniques from [49], this protocol can be used to obtain efficient protocols for secure linear algebra which make a black-box use of homomorphic encryption and achieve simulation-based security against malicious parties (improving over similar protocols with security against *covert* adversaries [3] recently presented in [49]).

All of our protocols are in fact *UC-secure* in the OT-hybrid model and can be generalized to *multiparty* computation with an arbitrary number of malicious

[2] The precise intractability assumption we use is similar in flavor to an assumption used in [50] for evaluating polynomials of degree $d \geq 2$. With a suitable choice of parameters, our assumption is implied by a natural pseudorandomness variant of the assumption from [50], discussed in [46]. The assumption does not seem to be affected by the recent progress on list-decoding Reed-Solomon codes and their variants [37,14,6,54].

parties. The security of the protocols also holds against *adaptive* adversaries, assuming that honest parties may erase data. (This is weaker than the standard notion of adaptive security [10] which does not rely on data erasure.) The *round complexity* of all the protocols is a constant multiple of the multiplicative depth of C.

From the OT-hybrid model to the plain model. An advantage of presenting our protocols in the OT-hybrid model is that they can be instantiated in a variety of models and under a variety of assumptions. For instance, using UC-secure OT protocols from [55,25], one can obtain efficient UC-secure instances of our protocols in the CRS model. In the stand-alone model, one can implement these OTs by making a black-box use of homomorphic encryption [42]. Thus, our protocols which make a black-box use of homomorphic encryption do not need to employ an additional OT primitive in the stand-alone model.

We finally note that our protocols require only $O(k)$ OTs with security in the malicious model, independently of the circuit size; the remaining OT invocations can all be implemented in the semi-honest model, which can be done very efficiently using the technique of [41]. Furthermore, all the "cryptographic" work for implementing the OTs can be done off-line, before any inputs are available. We expect that in most natural instances of large-scale secure arithmetic computation, the cost of realizing the OTs will not form an efficiency bottleneck.

Extensions. While we explicitly consider here only stateless arithmetic circuits, this model (as well as our results) can be readily generalized to allow stateful, reactive arithmetic computations whose secret state evolves by interacting with the parties.[3]

As it turns out, reactive arithmetic computations are useful not only for the obvious purpose of implementing stateful functionalities, but also, somewhat surprisingly, for enriching the (non-reactive) arithmetic computation model. They can be used to obtain efficient secure realizations of several "non-arithmetic" manipulations of the state, including decomposing a ring element into its bit-representation, equality testing, inversion, comparison, exponentiation, and others [21,59]. These reductions enhance the power of the basic arithmetic model, and allow protocols to efficiently switch from one representation to another in computations that involve both boolean and arithmetic operations.

2 Preliminaries

Black-box rings and fields. A probabilistic oracle R is said to be a valid implementation of a finite ring R if it behaves as follows: it takes as input one of the commands add, $\mathsf{subtract}$, $\mathsf{multiply}$, sample and two m bit "element identifiers" (or none, in the case of sample), and returns a single m bit string. There is a one-to-one mapping $\mathsf{label} : R \hookrightarrow \{0,1\}^m$ such that for all $x, y \in R$

[3] An ideal functionality which formally captures such general reactive arithmetic computations was defined in [24] (see also [59, Chapter 4]) and referred to as an *arithmetic black-box* (ABB). All of our protocols for arithmetic circuits can be naturally extended to realize the ABB functionality.

$\mathsf{R}(\mathsf{op}, \mathsf{label}(x), \mathsf{label}(y)) = \mathsf{label}(x *_R y)$ where op is one of add, $\mathsf{subtract}$ and $\mathsf{multiply}$ and $*_R$ is the ring operation $+, -$, or $\cdot$ respectively. When an input is not from the range of label, the oracle outputs $\perp$. (In a typical protocol, if a $\perp$ is ever encountered by an honest player, the protocol aborts.) The output of $\mathsf{R}(\mathsf{sample})$ is $\mathsf{label}(x)$ where x will be drawn uniformly at random from R. We will be interested in oracles of the kind that implements a *family* of rings, of varying sizes. Such a function should take an additional input id to indicate which ring it is implementing.

Definition 1. *A probabilistic oracle $\mathcal{R}$ is said to be a* concrete ring family *(or simply a* ring family*) if, for all strings id, the oracle $\mathcal{R}(\mathsf{id}, \cdot)$ (i.e., with first input being fixed to id), is an implementation of some ring. This concrete ring will be denoted by $\mathcal{R}_{\mathsf{id}}$.*

Note that so far we have not placed any computability requirement on the oracle; we only require a concrete mapping from ring elements to binary strings. However, when considering computationally secure protocols we will typically restrict the attention to "efficient" families of rings: we say $\mathcal{R}$ is a *computationally efficient ring family* if it is a ring family that can be implemented by a probabilistic polynomial time algorithm.

There are some special cases that we shall refer to:

1. Suppose that for all id, we have that $\mathcal{R}_{\mathsf{id}}$ is a ring with an identity for multiplication, 1. Then, we call $\mathcal{R}$ a *ring family with inverse* if in addition to the other operations, $\mathcal{R}(\mathsf{id}, \mathsf{one})$ returns $\mathsf{label}_{\mathsf{id}}(1)$ and $\mathcal{R}(\mathsf{id}, \mathsf{invert}, \mathsf{label}_{\mathsf{id}}(x))$ returns $\mathsf{label}_{\mathsf{id}}(x^{-1})$ if x is a unit (i.e., has a unique left- and right-inverse) and $\perp$ otherwise.
2. If $\mathcal{R}$ is a ring family with inverse such that for all id the ring $\mathcal{R}_{\mathsf{id}}$ is a field, then we say that $\mathcal{R}$ is a *field family*.
3. We call a ring family with inverse $\mathcal{R}$ a *pseudo-field family*, if for all id, all but negligible fraction of the elements in the ring $\mathcal{R}_{\mathsf{id}}$ are units.

Some special families of rings we will be interested in, other than finite fields, include rings of the form $\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$ for a composite integer m (namely, the ring of residue classes modulo m), and rings of matrices over a finite field or ring. With an appropriate choice of parameters, both of these families are in fact pseudo-fields. Note that a concrete ring family $\mathcal{R}$ for the rings of the form $\mathbb{Z}_m$ could use the binary representation of m as the input id; further the elements in $\mathbb{Z}_m$ could be represented as $\lceil \log m \rceil$-bit strings in a natural way. Of course, a different concrete ring family for the same ring can use a different representation.

Finally, for notational convenience we assume that the length of all element identifiers in $\mathcal{R}_{\mathsf{id}}$ is exactly $|\mathsf{id}|$. In particular, the ring $\mathcal{R}_{\mathsf{id}}$ has at most $2^{|\mathsf{id}|}$ elements.

Arithmetic circuits. We consider arithmetic circuits with gates labeled by add, $\mathsf{subtract}$, or $\mathsf{multiply}$. (In addition, for fields there is an additional constant gate one.) For a concrete ring family $\mathcal{R}$, we denote by $C^{\mathcal{R}}$ the mapping which

takes an id and a vector of input identifiers and outputs the corresponding vector of output identifiers. In the context of multi-party computation, each input or output to such a circuit is annotated to indicate which party (or parties) it "belongs" to. Given such an annotated circuit C and a concrete ring family $\mathcal{R}$, we define the functionality $\mathcal{F}_C^{\mathcal{R}}$ to behave as follows:

– The functionality takes id as a common (public) input, and receives (private) inputs to C from each party. It then evaluates the function $C^{\mathcal{R}}(\mathsf{id}, \text{inputs})$ using access to $\mathcal{R}$, and provides the outputs to the parties.[4]

Protocols securely realizing arithmetic computations. We follow the standard UC-security framework [9]. Informally, a protocol π is said to securely realize a functionality $\mathcal{F}$ if there exists a PPT simulator Sim, such that for all (non-uniform PPT) adversaries Adv, and all (non-uniform PPT) environments Env which interact with a set of parties and an adversary, the following two scenarios are indistinguishable: the REAL interaction where the parties run the protocol π and the adversary is Adv; the IDEAL interaction where the parties communicate directly with the ideal functionality $\mathcal{F}$ and the adversary is $\mathsf{Sim}^{\mathsf{Adv}}$. Indistinguishability can either be statistical (in the case of unconditional security) or computational (in the case of computational security). All parties, the adversary, the simulator, the environment and the functionality get the security parameter k as implicit input. Polynomial time computation, computational or statistical indistinguishability and non-uniformity are defined with respect to this security parameter k. However, since we don't impose an a-priori bound on the size of the inputs received from the environment as a function of k, the running time of honest parties is bounded by a fixed polynomial in the total length of their inputs (rather than a fixed polynomial in k).

We distinguish between *static* corruption and *adaptive* corruption. In the latter case it also makes a difference whether the protocols can erase part of their state (so that a subsequent corruption will not have access to the erased information), or no erasure is allowed. Our final protocols will have security against adaptive corruption *with erasures*.

We shall consider protocols which make oracle access to a ring family $\mathcal{R}$. The standard security definition is adapted to this case by giving all algorithms (including the environment) oracle access to $\mathcal{R}$. For such a protocol we define its *arithmetic computation complexity* as the number of oracle calls to $\mathcal{R}$. Similarly the *arithmetic communication complexity* is defined as the number of ring-element labels in the communication transcript. The arithmetic computation (respectively communication) complexity of our protocols will dominate the other computation steps in the protocol execution (respectively, the number of other bits in the transcript). Thus, the arithmetic complexity gives a good measure of efficiency for our protocols.

[4] $\mathcal{F}_C^{\mathcal{R}}$ can take id as input from each party, and ensure that all the parties agree on the same id. Alternately, we can restrict to environments which provide the same common input id to all parties.

Note that while any computational implementation of the ring oracle necessarily requires the complexity to grow with the ring size, it is possible that the arithmetic complexity does not depend on the size of the ring at all.

We now define our main notion of secure arithmetic computation.

Definition 2. *Let C be an arithmetic circuit. A protocol π is said to be a* secure black-box realization of *C-evaluation for a given set of ring families if, for each $\mathcal{R}$ in the set,*

1. *$\pi^{\mathcal{R}}$ securely realizes $\mathcal{F}_C^{\mathcal{R}}$, and*
2. *the arithmetic (communication and computation) complexity of $\pi^{\mathcal{R}}$ is bounded by some fixed polynomial in k and $|\mathsf{id}|$ (independently of $\mathcal{R}$).*

In the case of unconditional security we will quantify over the set of *all* ring families, whereas in the case of computational security we will typically quantify only over computationally efficient rings or fields. In both cases, the efficiency requirement on π rules out the option of using a brute-force approach to emulate the ring oracle by a boolean circuit.

We remark that our constructions will achieve a stronger notion of security, as the simulator used to establish the security in item (1) above will not depend on $\mathcal{R}$. A bit more precisely, the stronger definition is quantified as follows: there exists a simulator such that for all adversaries, ring families, and environments, the ideal process and the real process are indistinguishable. For simplicity however we phrase our definition as above which does allow different simulators for different $\mathcal{R}$.

3 Arithmetic Computation with Passive Corruption

To construct a protocol for general arithmetic circuit evaluation over a black-box ring family $\mathcal{R}$, that is secure against passive (adaptive) corruption it is enough to realize the following functionality $\mathcal{F}_{\mathsf{pdt\text{-}shr}}$ (see [45] for more details). Let $R = \mathcal{R}_{\mathsf{id}}$, where id is an implicit common input.

- A sends $a \in R$ and B sends $b \in R$ to $\mathcal{F}_{\mathsf{pdt\text{-}shr}}$.
- $\mathcal{F}_{\mathsf{pdt\text{-}shr}}$ samples two random elements $z^A, z^B \in R$ such that $z^A + z^B = ab$, and gives z^A to A and z^B to B.

A well-known approach for securely realizing this functionality against passive corruption, using a homomorphic encryption scheme (if available), goes as follows:

- Bob generates a public/secret key-pair encryption scheme, and sends an encryption of b along with the public key.
- Alice picks a random element z^A in the ring. She then computes an encryption of $ab - z^A$ from the encryption b (and the public-key) and sends it to Bob.
- Bob decrypts this ciphertext and accepts it as z^B. The encryption scheme should ensure that even with the secret-key, Bob does not learn anything else about (a, z^A) from the message she receives from Alice.

Indeed when such a homomorphic encryption scheme is available this gives a protocol with security against passive corruption for this task. However, such schemes are known only for select families of rings, and further do not meet the goal of making only black-box access to the ring.

This basic approach can be extended to the black-box ring setting with the help of an OT channel to ensure part of the privacy: Instead of an encryption, Alice sends an encoding of a under an appropriate *erasure correcting code*, but with sufficient noise to hide a from Bob. Her "secret-key" is the information about which co-ordinates are noisy. The code should have homomorphic properties to let Bob create a noisy encoding of $ab + z^B$ from this. To ensure that Alice does not learn anything beyond $ab + z^B$, Bob does not send the resulting noisy codeword to Alice, but lets her use an OT channel to pick up only the non-noisy co-ordinates of the codeword.

This high-level description fits the approach taken by Naor and Pinkas [50] for the special case of Reed-Solomon codes. Our protocols in this section provide more general and more efficient instantiations of this approach, to realize the functionality $\mathcal{F}_{\mathsf{pdt\text{-}shr}}$ described above. In Section 3.1 we describe our encoding schemes, and in Section 3.2 we show how these encoding schemes can be used in protocols that realize $\mathcal{F}_{\mathsf{pdt\text{-}shr}}$ against passive corruption.

3.1 Noisy Encodings

We describe several noisy encoding schemes based on linear codes. All our encoding schemes are specified using a *code generation algorithm* $\mathcal{G}$, over a ring family $\mathcal{R}$. $\mathcal{G}$ is a randomized algorithm such that $\mathcal{G}^{\mathcal{R}}(\mathsf{id}, k)$ outputs (G, L, H), where G is an $n \times k$ generator matrix of a linear code over $\mathcal{R}_{\mathsf{id}}$ of length $n = n(k)$, L is a subset of $[n]$ of size $\ell(k)$ which specifies the set of coordinates which are *not* replaced by noise, and H is another matrix which is used to facilitate efficient decoding. Here k is the security parameter as well as the code dimension, and $n(k)$ (code length) and $\ell(k)$ (number of coordinates *without* noise) are parameters of $\mathcal{G}$. In our instantiations n will be a constant multiple of k and in most cases we will have $\ell = k$.

Let $\mathcal{R}$ be a ring family. Given $\mathcal{G}$, a parameter $t(k) \leq k$ (number of ring elements to be encoded, $t = 1$ by default), and $x \in \mathcal{R}_{\mathsf{id}}^t$, we define a distribution $\mathcal{E}^{\mathcal{R}}_{(\mathcal{G},t)}(\mathsf{id}, k, x)$ as that of the public output in the following encoding process:

- *Encoding* $\mathsf{Encode}^{\mathcal{R}}_{(\mathcal{G},t)}(\mathsf{id}, k, x)$*:*
  - Input: $x = (x_1, \ldots, x_t) \in R^t$, where $R = \mathcal{R}_{\mathsf{id}}$ and $t = t(k)$.
  - Let $(G, L, H) \leftarrow \mathcal{G}^{\mathcal{R}}(\mathsf{id}, k)$
  - Pick a random vector $u \in R^k$ conditioned on $u_i = x_i$ for $i = 1, \ldots, t$ (i.e., u is x padded with $k - t$ random elements). Compute $Gu \in R^n$.
  - Let $v = Gu + e$, where $e \leftarrow R^n$ is drawn uniformly random conditioned on $e_i := 0$ for $i \in L$.
  - Let the private output be (G, L, H, v) and the public output be (G, v).

The matrix H is not used in the encoding above, but will be useful towards efficient decoding. In our main instantiations H can be readily derived from G

and L. We include H explicitly in the outcome of $\mathcal{G}$, because in some cases it is possible to obtain efficiency gains if (G, H, L) are sampled together.

Below we describe four instantiations of the above encoding scheme. The respective code generation algorithms are denoted by $\mathcal{G}_{\mathsf{Stat}}$, $\mathcal{G}_{\mathsf{Ring}}$, $\mathcal{G}_{\mathsf{Rand}}$, and $\mathcal{G}_{\mathsf{RS}}$. The first three use $t = 1$, i.e., a single ring element is encoded in a noisy codeword, and the last one allows $t(k)$ to be constant fraction of k, say $k/2$. The first three schemes allow homomorphic operations of multiplication and addition of the encoded element with an unencoded element. The last one allows coordinate wise multiplication and addition of t-long vectors. The last two require the ring family to be a field family.

The first encoding scheme has a statistical hiding property, whereas the others depend on computational assumptions for their hiding property. The assumption, in these three cases, is as follows:

Assumption 1 (Generic version, for a given $\mathcal{G}$, $\mathcal{R}$ and $t(k)$). *For all sequences $\{(\mathsf{id}_k, x_k, y_k)\}_k$ such that $x_k, y_k \in \mathcal{R}^{t(k)}_{\mathsf{id}_k}$, the ensembles $\{\mathcal{E}^{\mathcal{R}}_{(\mathcal{G},t)}(\mathsf{id}_k, k, x_k)\}_k$ and $\{\mathcal{E}^{\mathcal{R}}_{(\mathcal{G},t)}(\mathsf{id}_k, k, y_k)\}_k$ are computationally indistinguishable (by any $\mathrm{poly}(k)$-size nonuniform distinguisher).*

Statistically hiding encoding. Our statistically hiding encoding mixes an additive secret sharing of x with an equal number of uniformly random ring elements. Following is a more precise description of the encoding algorithm $\mathcal{G}_{\mathsf{Stat}}$ which fits into the above general framework.

- Let $R = \mathcal{R}_{\mathsf{id}}$. Let $n = 2m$ where $m = \log_2 |R| + k$.
- Let A_0 be the $m \times m$ matrix with 1 along the main diagonal and -1 along the rest of the first row.[5] Let G be the fixed $2m \times m$ matrix $G_0 = \left[\begin{smallmatrix} A_0 \\ A_0 \end{smallmatrix}\right]$.
- Define L as follows. Let $L = \{a_1, \ldots, a_m\}$ where $a_i = i$ or $m + i$ uniformly at random. (That is a_i indices the i-th row in one of the two copies of A_0.)
- Note that $G|_L = A_0$. H has 1 along the main diagonal and the first row, so that $HG|_L = I$.

The encoding of x is the vector $v = G_0 u + e$, where u is a random vector with $u_1 = x$ and e is a random noise vector with $e_i = 0$ for $i \in L$; v is then simply a random vector conditioned on $\sum_{i \in L} v_i = x$. This simple encoding has the useful property that it statistically hides x when the decoding information L is not provided. In the full version [45], we prove this fact using the Leftover Hash Lemma [39] (similarly to previous uses of this lemma in [40,43]).

Lemma 1. *For any $\mathcal{R}$, id, and $x \in \mathcal{R}_{\mathsf{id}}$, the statistical distance between the distribution of $\mathcal{E}^{\mathcal{R}}_{(\mathcal{G}_{\mathsf{Stat}},1)}(\mathsf{id}, k, x)$ and (G_0, v), where v is drawn uniformly from $\mathcal{R}^{2m}_{\mathsf{id}}$, is $2^{-\Omega(k)}$.*

We note that in light of efficient algorithms for low-density instances of subset sum, one cannot hope to obtain significant efficiency improvements by choosing a smaller value of m and settling for computational security.

[5] Here it is not necessary to assume that the ring has a multiplicative identity. In computing the matrix product, $1.a$ and $-1.a$ stand for a and $-a$.

Ring code based instantiation. Our next encoding scheme also uses $t = 1$, and works with any arbitrary ring family. It differs from the previous encoding scheme by not requiring n to depend on $|R|$; instead we fix $n = 2k$. The code generation algorithm, denoted by $\mathcal{G}_{\mathsf{Ring}}$, is very similar to $\mathcal{G}_{\mathsf{Stat}}$, except that $G = \left[\begin{smallmatrix} A \\ B \end{smallmatrix}\right]$, where A and B are two random $k \times k$ upper triangular matrices with 1 along the main diagonal. L is the same as before (using k instead of m). Note that $G|_L$ is an upper triangular matrix with 1 in the main diagonal. It is easy to compute an upper triangular matrix H (also with 1 in the main diagonal) using only the ring operations on elements in $G|_L$ such that $HG|_L = I$.

The hiding property is no longer statistical, but is a consequence of Assumption 1, instantiated with $\mathcal{G}_{\mathsf{Ring}}$ and $t = 1$. $\mathcal{G}_{\mathsf{Ring}}$ could be modified to use more than two matrices A and B, to make the resulting assumption weaker, at the expense of increasing n. In the full version we give an alternative to $\mathcal{G}_{\mathsf{Ring}}$ which relies on a random walk in the special linear group.

Random code based instantiation. Our next instantiation of the generic encoding, again with $t = 1$, uses a code generation algorithm $\mathcal{G}_{\mathsf{Rand}}$ based on a random linear code. It restricts the ring family to be a field family (or a pseudo-field family) $\mathcal{F}$. But this instantiation of Assumption 1 is a more standard assumption, which can be reduced to the hardness of *decoding* a random linear code when the field is small. $\mathcal{G}^{\mathcal{F}}_{\mathsf{Rand}}$ works as follows:

- Let $F = \mathcal{F}_{\mathsf{id}}$ and $n = 2k$.
- Pick a random $n \times k$ matrix $G \leftarrow F^{n \times k}$.
- Pick a random subset $L \subseteq [n]$, $|L| = k$, such that the $k \times k$ submatrix $G|_L$ is non-singular, where $G|_L$ consists of those rows in G whose indices are in L.
- Let $H = G|_L^{-1}$.

These three encoding schemes encode a single element in the ring (or field) R. They allow the following homomorphic operation: given an encoding of $x \in R$, namely $v = Gu + e$ where $u_1 = x$, for any $a, z \in R$, an encoding of $ax + z$ (with the same non-noisy co-ordinates) can be computed as $av + Gw$, where w is a random vector with $w_1 = z$. Further they all have the hiding property that after such a homomorphic operation, the non-noisy co-ordinates of the resulting encoding reveals nothing beyond the value $ax + z$. Finally, a noisy encoding $v \in R^n$ of $x \in R$ can be decoded by taking the first coordinate of $Hv|_L$.

Reed-Solomon code based instantiation. In our final instantiation of the generic encoding, we will let $t(k)$ be a constant fraction of k, say $t = k/2$. This variant of the construction exploits a stronger homomorphic property of Reed-Solomon codes, which was previously exploited in [30]. The code generation algorithm $\mathcal{G}_{\mathsf{RS}}$ uses $n = ck$, for a sufficiently large constant[6] $c > 4$. For a field $F = \mathcal{F}_{\mathsf{id}}$ the $n \times k$ matrix G is a linear transformation that extrapolates a degree $k - 1$ polynomial, given by its value at k randomly chosen points ζ_i in the field

[6] We require $c > 4$ so that Assumption 2(c) will not be broken by known list-decoding algorithms for Reed-Solomon codes [37]. Letting $c = 8$ may be a safe choice, with larger values of c being more conservative.

F, to n other randomly chosen evaluation points ϑ_i. (All ζ_i and ϑ_i are distinct,[7] and can be thought of as specifying G.) The non-noisy coordinates $L \subseteq [n]$ are chosen at random, where $|L| = 2k-1$ to allow reconstructing polynomials of degree $2(k-1)$.

This encoding allows the following homomorphic operation: given an encoding of $x \in F^t$, namely $v = Gu + e$ where $u_i = x_i$ for $i = 1, \ldots, t$, for any $a, z \in F^t$, an encoding of $ax + z$ (where ax denotes coordinate-wise multiplication) can be computed as $pv + w$, where $p, w \in F^n$ are the values of random polynomials of degree k and $2k$ respectively at the n evaluation points ϑ_i, which evaluate to a and z respectively in the first t of the k points ζ_i. Note that the resulting vector encodes (with noise) a degree $2k$ polynomial.

Instantiations of Assumption 1. Each of the above instantiations of the encoding leads to a corresponding instantiation of Assumption 1. For the sake of clarity we collect these assumptions below.

Assumption 2 *(a)* **[For $\mathcal{G}_{\mathsf{Rand}}$, with $t(k) = 1$].** *For every computationally efficient field family $\mathcal{F}$ and sequence $\{(\mathsf{id}_k, x_k, y_k)\}_k$ such that $x_k, y_k \in \mathcal{F}_{\mathsf{id}_k}$, the ensembles $\{\mathcal{E}^{\mathcal{F}}_{(\mathcal{G}_{\mathsf{Rand}},1)}(\mathsf{id}_k, k, x_k)\}_k$ and $\{\mathcal{E}^{\mathcal{F}}_{(\mathcal{G}_{\mathsf{Rand}},1)}(\mathsf{id}_k, k, y_k)\}_k$ are computationally indistinguishable.*

(b) **[For $\mathcal{G}_{\mathsf{Ring}}$, with $t(k) = 1$].** *For every computationally efficient ring family $\mathcal{R}$ and sequence $\{(\mathsf{id}_k, x_k, y_k)\}_k$ such that $x_k, y_k \in \mathcal{R}_{\mathsf{id}_k}$, the ensembles $\{\mathcal{E}^{\mathcal{F}}_{(\mathcal{G}_{\mathsf{Ring}},1)}(\mathsf{id}_k, k, x_k)\}_k$ and $\{\mathcal{E}^{\mathcal{F}}_{(\mathcal{G}_{\mathsf{Ring}},1)}(\mathsf{id}_k, k, y_k)\}_k$ are computationally indistinguishable.*

(c) **[For $\mathcal{G}_{\mathsf{RS}}$, with $t(k) = k/2$].**[8] *Let $t(k) = k/2$. For every computationally efficient field family $\mathcal{F}$ and sequence $\{(\mathsf{id}_k, x_k, y_k)\}_k$ such that $x_k, y_k \in \mathcal{F}^{t(k)}_{\mathsf{id}_k}$, the ensembles $\{\mathcal{E}^{\mathcal{F}}_{(\mathcal{G}_{\mathsf{RS}},t(k))}(\mathsf{id}_k, k, x_k)\}_k$ and $\{\mathcal{E}^{\mathcal{F}}_{(\mathcal{G}_{\mathsf{RS}},t(k))}(\mathsf{id}_k, k, y_k)\}_k$ are computationally indistinguishable.*

3.2 Product-Sharing Secure against Passive Corruption

Below we list the protocols for securely realizing $\mathcal{F}_{\mathsf{pdt\text{-}shr}}$ that we obtain from the noisy encodings above. They have increasing efficiency, but use stronger assumptions. These protocols use only black-box access to the ring (or field), and are in the OT-hybrid model. The protocols follow the pattern described at the beginning of Section 3. But this achieves security only against *static* corruption. In the full version [45] we show how to transform them into protocols that are secure against adaptive passive corruption, with erasures. In brief, in the new protocol, first the original protocol is run on random inputs, and then its working memory is deleted, and finally the real inputs and the outcome of the original protocol are used to complete the protocol.

[7] This requires to ensure that $|F| > n + k$. If id does not satisfy this requirement the algorithm uses a sufficiently large extension field of F.

[8] We can make the assumption weaker by choosing smaller values of t, or larger values of n in $\mathcal{G}_{\mathsf{RS}}$.

The different protocols are as follows:

– Protocol ρ^{OT} (with statistical security): this protocol uses the statistically hiding encoding scheme based on $\mathcal{G}_{\mathsf{Stat}}$, and achieves statistical security.

– Protocol σ^{OT}: this protocol uses the computationally hiding encoding scheme based on $\mathcal{G}_{\mathsf{Rand}}$ (for fields) or $\mathcal{G}_{\mathsf{Ring}}$. Security follows from Assumption 2(a) or (b), respectively.

– Protocol τ^{OT} (using packed encoding): this protocol uses the noisy encoding scheme with the code generation algorithm $\mathcal{G}_{\mathsf{RS}}$. It realizes multiple ($t = k/2$) parallel sessions of $\mathcal{F}_{\mathsf{pdt\text{-}shr}}$. Security follows from Assumption 2(c).

4 Arithmetic Computation with Active Corruption

In [44] it is shown how to obtain a UC-secure protocol in the OT-hybrid model for any two-party functionality $\mathcal{F}$ against active corruption by making a *black-box* use of the following two ingredients:

1. An "outer protocol" for $\mathcal{F}$ which employs k auxiliary parties (servers); this protocol should be UC-secure against active corruption provided that only some constant fraction of the servers can be (adaptively) corrupted.

2. An "inner protocol" for a reactive two-party functionality corresponding to each server in the outer protocol. In contrast to the outer protocol, this protocol only needs to be secure against *passive* (adaptive) corruption. The inner protocol is allowed to be in the OT-hybrid model and to have memory erasures.

Below we summarize the results we obtain by combining appropriate choices for the outer protocol with the inner protocols from Section 3. All these results can be readily extended to the multi-party setting as well, where the complexity grows polynomially with the number of parties; see [45] for details. All of the protocols provide adaptive security with erasures.

Combining the protocol from [19] (which makes a black-box use of an arbitrary ring) as the outer protocol with ρ^{OT} as the inner protocol, we get:

Theorem 1 (Unconditionally Secure Protocol). *For any arithmetic circuit C, there exists a protocol Π in the OT-hybrid model that is a secure black-box realization of C-evaluation for the set of all ring families. The security holds unconditionally against computationally unbounded adversaries and environments.*

The *arithmetic* communication complexity of the protocol ρ^{OT}, and hence that of the above protocol, grows linearly with (a bound on) $|\log \mathcal{R}_{\mathsf{id}}|$. To obtain a computationally secure protocol whose arithmetic communication complexity is independent of the ring, we can replace ρ^{OT} by σ^{OT} (with $\mathcal{G}_{\mathsf{Rand}}$ as the code generation scheme) in the previous construction:

Theorem 2. *Suppose that Assumption 2(a) holds. Then, for every arithmetic circuit C, there exists a protocol Π in the OT-hybrid model that is a secure black-box realization of C-evaluation for the set of all computationally efficient field families $\mathcal{F}$. Further, the arithmetic complexity of Π is* $\mathrm{poly}(k) \cdot |C|$, *independent of $\mathcal{F}$ or* id.

Using $\mathcal{G}_{\mathsf{Ring}}$ instead of $\mathcal{G}_{\mathsf{Rand}}$, this result extends to all computationally efficient ring families:

Theorem 3. *Suppose that Assumption 2(b) holds. Then, for every arithmetic circuit C, there exists a protocol Π in the OT-hybrid model that is a secure black-box realization of C-evaluation for the set of all computationally efficient ring families $\mathcal{R}$. Further, the arithmetic complexity of Π is $\mathrm{poly}(k) \cdot |C|$, independent of $\mathcal{R}$ or id.*

Finally, to obtain our most efficient protocol we use τ^{OT} (with $n = O(k)$ and $t = \Omega(k)$) as the inner protocol. The outer protocol is a variant of the protocol from [22] in which the computational complexity is optimized using an idea from [36] (see [45] for a description). To get the computational complexity specified below, the size of the field should be super-polynomial in the security parameter. (The communication complexity does not depend on this assumption.)

Theorem 4. *Suppose that Assumption 2(c) holds. Then, for every arithmetic circuit C, there exists a protocol Π in the OT-hybrid model with the following properties. The protocol Π is a secure black-box realization of C-evaluation for the set of all computationally efficient field families $\mathcal{F}$, with respect to all computationally bounded environments for which $|\mathcal{F}_{\mathsf{id}}|$ is super-polynomial in k. The arithmetic communication complexity of Π is $O(|C| + k \cdot \mathsf{depth}(C))$, where $\mathsf{depth}(C)$ denotes the depth of C, and its arithmetic computation complexity is $O(\log^2 k) \cdot (|C| + k \cdot \mathsf{depth}(C))$. Its round complexity is $O(\mathsf{depth}(C))$.*

By using a suitable choice of fields and evaluation points for the Reed-Solomon encoding, and under a corresponding specialization of Assumption 2(c), the computational overhead of the above protocol can be reduced from $O(\log^2 k)$ to $O(\log k)$. (In this variant we do not attempt to make a black-box use of the underlying field and rely on the standard representation of field elements.)

4.1 Protocols from Homomorphic Encryption

So far we considered protocols using only black-box access to a ring. If we further assume a black-box access[9] to a homomorphic encryption scheme over the ring, there are simple protocols for $\mathcal{F}_{\mathsf{pdt\text{-}shr}}$ secure against passive adversaries. These can then be used instead of the protocols from Section 3.2 in the above constructions.

We are interested in homomorphic encryptions over rings that support addition of two encrypted elements, and multiplication of an encrypted element by an unencrypted element. There are two kinds of such schemes. The more versatile kind — which we shall call a *controlled-ring* scheme — allows one to specify id during the key-generation phase, and then allows operations on elements in $\mathcal{R}_{\mathsf{id}}$, where $\mathcal{R}$ is the ring family associated with the scheme. Candidates for such schemes are the classic Goldwasser-Micali encryption scheme [35] (for which the ring family consists of the single ring $\mathbb{Z}_2$) and Benaloh's scheme [5] (for which

[9] When saying that a construction makes a black-box use of a homomorphic encryption primitive, we refer to the notion of a fully black-box reduction as defined in [58].

the ring family consists of rings $\mathbb{Z}_p$ where p is a polynomially bounded prime number). Any such $\mathcal{R}$-homomorphic encryption scheme can be used as a black-box to obtain a homomorphic encryption scheme for the ring family of square matrices over $\mathcal{R}$ (where the matrix size n is specified by id).

Theorem 5. *For every arithmetic circuit C, there exists a protocol Π in the* OT*-hybrid model, such that for every ring family $\mathcal{R}$, the protocol $\Pi^{\mathcal{R}}$ securely realizes $\mathcal{F}_C^{\mathcal{R}}$ by making a* black-box *use of any controlled-ring homomorphic encryption for $\mathcal{R}$. The number of invocations of the encryption scheme is* $\mathrm{poly}(k) \cdot |C|$*, independent of $\mathcal{R}$ or* id.

Note that the protocol in the above theorem, when instantiated with the ring of $n \times n$ matrices over $\mathbb{Z}_p$, has communication complexity $poly(k) \cdot |C| \cdot n^2$. Combined with [49], this yields constant-round protocols for secure linear algebra which make a black-box use of homomorphic encryption and whose communication complexity is nearly linear in the input size.

For the case of fields, we obtain the following more efficient version of the result by using the same outer protocol as used in Theorem 4:

Theorem 6. *For every arithmetic circuit C, there exists a protocol Π in the* OT*-hybrid model, such that for every field family $\mathcal{F}$, the protocol $\Pi^{\mathcal{F}}$ securely realizes $\mathcal{F}_C^{\mathcal{F}}$ by making a* black-box *use of any controlled-ring homomorphic encryption for $\mathcal{F}$. The security holds against adaptive corruption with erasures. Further, Π makes $O(|C| + k \cdot \mathsf{depth}(C))$ invocations of the encryption scheme, and the communication complexity is dominated by sending $O(|C| + k \cdot \mathsf{depth}(C))$ ciphertexts.*

Homomorphic encryption schemes like the Paillier cryptosystem [53] are homomorphic with respect to the ring $\mathbb{Z}_N$, where N is a randomly chosen product of two large primes chosen at the time of key generation; N cannot be specified ahead of time. We call such a scheme an "uncontrolled ring" homomorphic encryption scheme. Using standard techniques computation over $\mathbb{Z}_M$ for an *a priori* fixed modulus M can be securely reduced to computation over $\mathbb{Z}_N$ where N is a sufficiently large, dynamically chosen modulus (see [45] for more details). We obtain the following results:

Theorem 7. *Let $\mathcal{R}$ be the ring family where $\mathcal{R}_{\mathsf{id}}$ is the standard representation of the ring $\mathbb{Z}_{\mathsf{id}}$. For every arithmetic circuit C there exists a black-box construction of a protocol Π in the* OT*-hybrid model from any uncontrolled-ring homomorphic encryption for $\mathcal{R}$, such that Π is a secure realization of C-evaluation for $\mathcal{R}$. The number of invocations of the encryption scheme is* $\mathrm{poly}(k) \cdot |C|$*, independent of* id*, and the communication complexity is dominated by* $\mathrm{poly}(k) \cdot |C|$ *ciphertexts. During the protocol, the ring size parameter fed to the encryption scheme by honest parties is limited to $k' = O(k + |\mathsf{id}|)$.*

If, further, the ring over which C should be computed is restricted to be a field, there exists a protocol as above which makes $O(|C| + k \cdot \mathsf{depth}(C))$ invocations of the encryption scheme, and where the communication complexity is dominated by sending $O(|C| + k \cdot \mathsf{depth}(C))$ ciphertexts.

The second part of the above theorem also applies to the case of arithmetic computation over pseudo-fields. Furthermore, it can be generalized to the ring of $n \times n$ matrices, which when used with constructions of uncontrolled-ring $\mathbb{Z}_N$-homomorphic encryption schemes from the literature [53,23] would yield arithmetic protocols for matrices over large rings whose complexity grows quadratically with n.

We finally note that in the *stand-alone* model, the OT oracle in the above protocols can be realized by making a black-box use of the homomorphic encryption primitive without affecting the asymptotic number of calls to the primitive. This relies on the black-box construction from [42] and the fact that only $O(k)$ OTs need to be secure against active corruption. Thus, the above theorems hold also in the plain, stand-alone model (as opposed to the OT-hybrid UC-model).

Acknowledgments. We thank Jens Groth, Venkatesan Guruswami, Farzad Parvaresh, Oded Regev, and Ronny Roth for helpful discussions.

References

1. Abadi, M., Feigenbaum, J.: Secure circuit evaluation. J. Cryptology 2(1), 1–12 (1990)
2. Algesheimer, J., Camenisch, J., Shoup, V.: Efficient computation modulo a shared secret with application to the generation of shared safe-prime products. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 417–432. Springer, Heidelberg (2002)
3. Aumann, Y., Lindell, Y.: Security against covert adversaries: Efficient protocols for realistic adversaries. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 137–156. Springer, Heidelberg (2007)
4. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: STOC 1988, pp. 1–10 (1988)
5. Benaloh, J.: Verifiable Secret-Ballot Elections. PhD thesis, Department of Computer Science, Yale University (1987)
6. Bleichenbacher, D., Kiayias, A., Yung, M.: Decoding interleaved reed-solomon codes over noisy channels. Theor. Comput. Sci. 379(3), 348–360 (2007)
7. Bogetoft, P., Christensen, D.L., Damgard, I., Geisler, M., Jakobsen, T., Krøigaard, M., Nielsen, J.D., Nielsen, J.B., Nielsen, K., Pagter, J., Schwartzbach, M., Toft, T.: Multiparty computation goes live. Cryptology ePrint Archive, Report 2008/068
8. Boneh, D., Franklin, M.K.: Efficient generation of shared RSA keys. J. ACM 48(4), 702–722 (2001); Earlier version in Crypto 1997
9. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067 (2005)
10. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: STOC 1996, pp. 639–648 (1996)
11. Canetti, R., Ishai, Y., Kumar, R., Reiter, M.K., Rubinfeld, R., Wright, R.N.: Selective private function evaluation with applications to private statistics. In: PODC 2001, pp. 293–304 (2001)
12. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party computation. In: STOC 2002, pp. 494–503 (2002)

13. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols. In: STOC 1988, pp. 11–19 (1988)
14. Coppersmith, D., Sudan, M.: Reconstructing curves in three (and higher) dimensional space from noisy data. In: STOC 2003, pp. 136–142 (2003)
15. Cramer, R., Damgård, I.: Secure distributed linear algebra in a constant number of rounds. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 119–136. Springer, Heidelberg (2001)
16. Cramer, R., Damgård, I.B., Maurer, U.M.: General secure multi-party computation from any linear secret-sharing scheme. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 316–334. Springer, Heidelberg (2000)
17. Cramer, R., Damgård, I.B., Nielsen, J.B.: Multiparty computation from threshold homomorphic encryption. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 280–299. Springer, Heidelberg (2001)
18. Cramer, R., Fehr, S.: Optimal black-box secret sharing over arbitrary abelian groups. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 272–287. Springer, Heidelberg (2002)
19. Cramer, R., Fehr, S., Ishai, Y., Kushilevitz, E.: Efficient multi-party computation over rings. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 596–613. Springer, Heidelberg (2003)
20. Cramer, R., Kiltz, E., Padró, C.: A note on secure computation of the Moore-Penrose pseudoinverse and its application to secure linear algebra. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 613–630. Springer, Heidelberg (2007)
21. Damgård, I., Fitzi, M., Kiltz, E., Nielsen, J.B., Toft, T.: Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 285–304. Springer, Heidelberg (2006)
22. Damgård, I., Ishai, Y.: Scalable secure multiparty computation. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 501–520. Springer, Heidelberg (2006)
23. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 79–95. Springer, Heidelberg (2002)
24. Damgård, I., Nielsen, J.B.: Universally composable efficient multiparty computation from threshold homomorphic encryption. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 247–264. Springer, Heidelberg (2003)
25. Damgård, I., Nielsen, J.B., Orlandi, C.: Essentially optimal universally composable oblivious transfer. In: ICISC 2008 (2008)
26. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, Heidelberg (1990)
27. Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. Commun. ACM 28(6), 637–647 (1985)
28. Frankel, Y., MacKenzie, P.D., Yung, M.: Robust efficient distributed rsa-key generation. In: STOC 1998, pp. 663–672 (1998)
29. Franklin, M.K., Haber, S.: Joint encryption and message-efficient secure computation. J. Cryptology 9(4), 217–232 (1996)
30. Franklin, M.K., Yung, M.: Communication complexity of secure computation (extended abstract). In: STOC 1992, pp. 699–710 (1992)
31. Freedman, M.J., Nissim, K., Pinkas, B.: Efficient private matching and set intersection. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 1–19. Springer, Heidelberg (2004)
32. Gilboa, N.: Two party RSA key generation. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 116–129. Springer, Heidelberg (1999)

33. Goldreich, O.: Foundations of Cryptography: Basic Applications. Cambridge University Press, Cambridge (2004)
34. Goldreich, O., Micali, S., Wigderson, A.: How to play ANY mental game. In: STOC 1987, pp. 218–229 (1987); See [ch. 7] for more details.
35. Goldwasser, S., Micali, S.: Probabilistic encryption. J. Comput. Syst. Sci. 28(2), 270–299 (1984); Preliminary version in STOC 1982
36. Groth, J.: Linear algebra with sub-linear zero-knowledge arguments (manuscript, 2008)
37. Guruswami, V., Sudan, M.: Improved decoding of reed-solomon and algebraic-geometry codes. IEEE Trans. Inf. Theory 45(6), 1757–1767 (1999)
38. Hazay, C., Lindell, Y.: Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 155–175. Springer, Heidelberg (2008)
39. Impagliazzo, R., Levin, L.A., Luby, M.: Pseudo-random generation from one-way functions (extended abstract). In: STOC 1989, pp. 12–24 (1989)
40. Impagliazzo, R., Naor, M.: Efficient cryptographic schemes provably as secure as subset sum. J. Cryptology 9(4), 199–216 (1996)
41. Ishai, Y., Kilian, J., Nissim, K., Petrank, E.: Extending oblivious transfers efficiently. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 145–161. Springer, Heidelberg (2003)
42. Ishai, Y., Kushilevitz, E., Lindell, Y., Petrank, E.: Black-box constructions for secure computation. In: STOC 2006, pp. 99–108 (2006)
43. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Cryptography from anonymity. In: FOCS 2006, pp. 239–248 (2006)
44. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer - efficiently. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 572–591. Springer, Heidelberg (2008)
45. Ishai, Y., Prabhakaran, M., Sahai, A.: Secure arithmetic computation with no honest majority. Cryptology ePrint Archive, Report 2008/465 (2008)
46. Kiayias, A., Yung, M.: Cryptographic hardness based on the decoding of reed-solomon codes. IEEE Transactions on Information Theory 54(6), 2752–2769 (2008)
47. Kiltz, E., Mohassel, P., Weinreb, E., Franklin, M.K.: Secure linear algebra using linearly recurrent sequences. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 291–310. Springer, Heidelberg (2007)
48. Lindell, Y., Pinkas, B.: Privacy preserving data mining. J. Cryptology 15(3), 177–206 (2002); Earlier version in Crypto 2000
49. Mohassel, P., Weinreb, E.: Efficient secure linear algebra in the presence of covert or computationally unbounded adversaries. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 481–496. Springer, Heidelberg (2008)
50. Naor, M., Pinkas, B.: Oblivious polynomial evaluation. SIAM J. Comput. 35(5), 1254–1281 (2006); Earlier version in STOC 1999
51. Naor, M., Pinkas, B., Sumner, R.: Privacy preserving auctions and mechanism design. In: ACM Conference on Electronic Commerce 1999, pp. 129–139 (1999)
52. Nissim, K., Weinreb, E.: Communication efficient secure linear algebra. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 522–541. Springer, Heidelberg (2006)
53. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
54. Parvaresh, F., Vardy, A.: Correcting errors beyond the guruswami-sudan radius in polynomial time. In: FOCS 2005, pp. 285–294 (2005)

55. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (2008)
56. Poupard, G., Stern, J.: Generation of shared RSA keys by two parties. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 11–24. Springer, Heidelberg (1998)
57. Rabin, M.: How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory (1981)
58. Reingold, O., Trevisan, L., Vadhan, S.P.: Notions of reducibility between cryptographic primitives. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 1–20. Springer, Heidelberg (2004)
59. Toft, T.: Primitives and Applications for Multi-party Computation. PhD thesis, Department of Computer Science, Aarhus University (2007)
60. Yao, A.C.: How to generate and exchange secrets. In: FOCS 1996, pp. 162–167 (1996)