The Round Complexity of Perfect MPC with Active Security and Optimal Resiliency
(Extended Abstract)

Benny Applebaum
School of Electrical Engineering
Tel-Aviv University
Tel-Aviv, Israel
benny.applebaum@gmail.com

Eliran Kachlon
School of Electrical Engineering
Tel-Aviv University
Tel-Aviv, Israel
elirn.chalon@gmail.com

Arpita Patra
Dept. of Computer Science and Automation
Indian Institute of Science
Bangalore, India
arpita@iisc.ac.in

***Abstract*—In STOC 1988, Ben-Or, Goldwasser, and Wigderson (BGW) established an important milestone in the fields of cryptography and distributed computing by showing that every functionality can be computed with perfect (information-theoretic and error-free) security at the presence of an active (aka Byzantine) rushing adversary that controls up to $n/3$ of the parties.**

We study the round complexity of *general* secure multiparty computation in the BGW model. Our main result shows that every functionality can be realized in only four rounds of interaction, and that some functionalities cannot be computed in three rounds. This completely settles the round-complexity of perfect actively-secure optimally-resilient MPC, resolving a long line of research.

Our lower-bound is based on a novel round-reduction technique that allows us to lift existing three-round lower-bounds for verifiable secret sharing to four-round lower-bounds for general MPC. To prove the upper-bound, we develop new round-efficient protocols for computing degree-2 functionalities over large fields, and establish the completeness of such functionalities. The latter result extends the recent completeness theorem of Applebaum, Brakerski and Tsabary (TCC 2018, Eurocrypt 2019) that was limited to the binary field.

***Keywords*-Information-Theoretic Cryptography, Cryptographic Protocols, Secure Computation, Round Complexity**

I. Introduction

The round complexity of interactive protocols is one of their most important efficiency measures. Consequently, a huge amount of research has been devoted towards characterizing the round complexity of various distributed tasks (e.g., Byzantine agreement [22], [23], [37], coin flipping [19], [38], zero-knowledge proofs [16], [29], verifiable secret sharing [27], [35] and secure multiparty computation [12], [14], [26], [44]) under different security models. In this paper, we study the round complexity of *general* secure multiparty computation (MPC) in the classical model of Ben-Or, Goldwasser, and Wigderson [13]. That is, we strive for *perfect (information-theoretic and error-free)* security at the presence of an *active* (aka Byzantine or malicious) rushing adversary that controls up to $n/3$ of the parties.

In more detail, in this setting n parties wish to compute a joint function f of their private inputs. We assume that parties can communicate over *secure point-to-point channels* and, in addition, have an access to a *broadcast* channel. An all-powerful, computationally-unbounded, adversary actively corrupts up to a bounded number t of the parties, and may instruct the corrupted parties to arbitrarily deviate from the protocol. Informally, *perfect* security essentially implies that the honest parties will always get a valid output, and there is a *zero* probability of cheating by the adversary.[1] We focus on the most challenging setting in which the adversary may corrupt up to $t = \lceil n/3 \rceil - 1$ parties, which is known to be the best achievable resiliency threshold in this setting [13], [41].

The discovery of perfect MPC is no less than remarkable. It provides everlasting security unconditionally without relying on unproven intractability assumptions. In one of the most seminal results in the area of cryptography and distributed computing, Ben-Or, Goldwasser, and Wigderson [13] and Chaum, Crépeau and Damgård [17] established the following completeness theorem.

Theorem 1.1 (Feasibility of perfect MPC): Every n-party functionality can be computed with perfect security against a computationally-unbounded adversary that actively corrupts up to $t < n/3$ of the parties.

The round complexity of information-theoretic MPC was extensively studied [1], [2], [4], [5], [9], [11], [25], [27], [28], [30]–[33], [35], [39], [42]. While it is known that perfect actively-secure MPC can be carried in a constant number of rounds [32], the exact complexity has remained open. In this paper, we resolve this question, and completely

The full version of this paper can be found in [7]. The first two authors are supported by the European Union's Horizon 2020 Programme (ERC-StG-2014-2020) under grant agreement no. 639813 ERC-CLC, and the Check Point Institute for Information Security. Arpita Patra would like to acknowledge financial support from SERB MATRICS (Theoretical Sciences) Grant 2020 and Google India AI/ML Research Award 2020.

[1] Apart from being a natural goal, perfect security provides important and useful security advantages over protocols that have a negligible probability of failure, see, e.g., [36].

2575-8454/20/$31.00 ©2020 IEEE
DOI 10.1109/FOCS46700.2020.00121

characterize the round complexity of perfect actively-secure MPC.

Theorem 1.2 (Round Complexity of perfect MPC): Four rounds are necessary and sufficient for general MPC with perfect active-security and optimal resiliency of $t < n/3$.

To prove the theorem, we develop new lower-bound and upper-bound techniques. Before providing a detailed account of our results and techniques, let us describe some of the most relevant previous results regarding the round-complexity of perfectly-secure MPC.

A. Previous Works

The classical approach for perfectly-secure MPC (e.g., Theorem 1.1) suffers from a large round complexity. Loosely speaking, the idea is to represent the function f as an arithmetic circuit C over some moderate-size finite field $\mathbb{F}$ of cardinality larger than n, and to distributively evaluate this circuit in a gate by gate manner. At the beginning, each party uses a *verifiable secret sharing* (VSS) sub-protocol [18] in order to share her input based on degree-t polynomials [43]. The parties then evaluate the circuit over their shares. Addition is performed locally with no interaction, whereas multiplication increases the degree of the underlying secret-sharing polynomial. To compensate this, after every level of multiplication gates the parties employ a "degree-reduction" step that is based again on VSS. Finally, the parties broadcast their shares for the output wires. The resulting round complexity is therefore $(d+1) \cdot R_{\text{vss}} + 1$ where d is the multiplicative depth of C and R_{vss} is the round complexity of the sharing a secret via perfect VSS with optimal resiliency.[2]

Several works have shown how to securely compute various functionalities in constant number of rounds and with small error probability (e.g., [9], [20]). In [31], [32], Ishai and Kushilevitz (IK) presented the *randomizing polynomials* methodology and showed that one can securely reduce, in a round-preserving way, the computation of an arbitrary functionality to the computation of a degree-3 functionality in which each output is a degree-3 polynomial in the inputs/randomness of the parties. This technique, combined with the aforementioned protocols, allows to securely compute an arbitrary functionality in a constant number of rounds. The overhead of the reduction is polynomial in the formula-size (or branching program size) of f, and so the resulting protocol has a polynomial complexity only for functions computable in $\mathbf{NC}^1$ or in (counting versions of) log-space. Similar limitations apply to all known information-theoretic constant-round protocols even in the case of statistical security against a passive adversary.

The discovery of constant-degree randomizing polynomials provides a tighter relation between the round complexity of VSS and the round complexity of general functionalities. Motivated by this connection, Gennaro et al. [27] studied the round-complexity of VSS, and proved that three rounds are necessary and sufficient for (the sharing phase of) perfect VSS with optimal resiliency of $t = \lceil n/3 \rceil - 1$. A large number of follow-up works have established other bounds on the round-complexity of VSS in different models (statistical and computational security), network setting and under more refined metrics (minimizing broadcast rounds) [8], [24], [34], [35], [39], [40]. Since the sharing functionality of VSS is by itself an MPC task, this yields a three-round lower-bound for securely computing a general functionality perfectly. The lower bound of 3 rounds is also implied from the result of [28] for any $t > 1$.

Very recently, Applebaum et al. [4], [5], inspired by breakthroughs in computational MPC [14], [26], presented a multiparty-variant of randomizing polynomials and used it to securely reduce any functionality f to the computation of some degree-2 functionality g. Unlike the IK constructions, this new completeness result is inherently limited to the binary field. That is, the target functionality g is a degree-2 mapping over $\mathbb{F}_2$. As a result, g cannot be directly computed via the polynomial-based protocols which require a field $\mathbb{F}$ of size at least $n+1$. This seemingly technical mismatch becomes a real issue in the setting of an active adversary. Naively, one can try to replace g with its "arithmetic" version G, which is a degree-2 functionality over $\mathbb{F}$ that agrees with g over binary inputs. However, a protocol for G does not translate into a protocol for g since the adversary may use non-binary inputs.[3] Applebaum et al. [5] were not able to overcome this difficulty in the general case, but were able to obtain new improved round-complexity upper-bounds for *weaker* security notions (e.g., where honest parties are allowed to "abort") or with *sub-optimal* resiliency threshold. Specifically, they constructed a three-round perfectly-secure protocol against an active adversary with threshold of $t < n/4$. The use of binary quadratic function significantly complicates the protocol and its analysis [5, Section 6.1, Appendix A]. For comparison, a similar protocol for quadratic functionalities over a large field can be trivially constructed based on existing techniques from [27].

[2] In fact, using more modern preprocessing tricks (e.g., due to [10], [21]) one can pay only a single round of interaction per multiplication level at the expense of distributing at the beginning shares of many multiplication triples (A_i, B_i, C_i) where A_i, B_i, C_i are random degree-t polynomials for which $A_i(0) \cdot B_i(0) = C_i(0)$. Since such a sharing can be implemented by a degree-2 functionality, this leads to a round complexity of $R_2 + d + 1$, where R_2 is the complexity of computing an arbitrary degree-2 polynomial. One can further collapse the initial sharing and the first level of multiplication into a single degree-2 functionality, that can be performed in parallel to the triple-generation step, and improve the round complexity to $R_2 + (d-1) + 1$. However, these optimizations do not help for quadratic functions which will be our focus here.

[3] This transformation actually works when the adversary is passive. Indeed, by using this route [4] constructed an optimal two-round MPC with *passive* perfect security at the presence of honest majority.

B. Our Results

We prove the first four-round lower-bound for perfectly-secure MPC.

Theorem 1.3 (3-rounds are insufficient for perfect-MPC): Let $n \geq 4$ and $n/3 \geq t \geq n/4$ be positive integers. Then there exists an n-party functionality f that cannot be computed in three rounds with perfect security and t-resiliency.

As an immediate corollary, we conclude that the three-round protocol of Applebaum et al. [5] that achieves a resiliency-threshold of $\lceil n/4 \rceil - 1$ cannot be improved, and that at least four rounds are necessary in order to achieve an optimal threshold. To prove the theorem, we introduce a new round-reduction technique, and use it to show that the number of rounds needed for general MPC is strictly larger than the number of rounds needed for VSS. (See Section II-A for an outline.) We complement Theorem 1.3 by proving the following upper-bound.

Theorem 1.4 (4-rounds are sufficient for perfect-MPC): Every n-party functionality can be computed in four rounds with perfect active-security and optimal resiliency of $t < n/3$. The complexity of the protocol is polynomial in n and in the formula-size (or branching program size) of f.

Taken together, Theorems 1.3 and 1.4 completely characterize the round complexity of perfectly-secure computation with optimal resiliency.

As usual in the context of constant-round information-theoretic MPC, our protocol is efficient only for $\mathbf{NC}^1$ or log-space functionalities. Nevertheless, even for general functions, for which our construction is inefficient, the result remains meaningful since the protocol resists computationally unbounded adversaries.

The protocol from Theorem 1.4 is proven to be perfectly-secure via a black-box non-rewinding simulator whose complexity is polynomial in the complexity of the adversary and the formula (or branching program) size of f. These features (which are common in the information-theoretic setting) have several useful corollaries. First, by [36], this implies that the protocol is also universally composable (UC) [15], that is, security is preserved even when many arbitrary protocols are run concurrently with our protocol. Moreover, when the protocol is efficiently computable (e.g., for $f \in \mathbf{NC}^1$), we can use public-key encryption to get rid of the private-channel assumption, at the expense of degrading perfect UC-security to computational UC-security. The resulting protocol can be implemented in the "authenticated channels model" without making use of any *set-up assumptions*!

The proof of Theorem 1.4 is based on two components. Our first ingredient is an extension of the degree-2 completeness result of [5] to large fields of characteristic 2.

Theorem 1.5 (Completeness of quadratic functions): Let f be an n-party functionality, and let $k \geq 1$ be an integer. Then there exists a non-interactive reduction from the task of securely computing f to the task of computing a degree-2 functionality over the field $\mathbb{F}_{2^k}$. The reduction preserves perfect, statistical and computational active-security and provides optimal resiliency.

This new completeness result can be used to significantly simplify some of the protocols from [5], [33]. We further believe that it will lead to new round-optimal protocols in other settings. The proof is outlined in Section II-C. (See also the full version of the paper [7] for a full statement of the theorem.) Theorem 1.5 is accompanied with a new four-round protocol for degree-2 functionalities.

Theorem 1.6 (Four-round protocol for quadratic functions): Every n-party degree-2 functionality over a finite field $\mathbb{F}$ of size at least n, can be securely computed in four rounds with perfect active-security and optimal resiliency of $t < n/3$.

Just like in the case of our lower-bound, the proof of Theorem 1.6 essentially establishes a (tight) relation between the round complexity of MPC and VSS– it shows that MPC for quadratic functions essentially needs only *one* additional round beyond what is needed for VSS. The proof (which is outlined in Section II-B and fully appears in the full version of this paper [7]) introduces several novel techniques.

II. Technical Overview

In the following subsections, we outline the main ideas behind the proofs of our results. (The reader is referred to the full version [7] for further details.)

A. Lower Bound

Our starting point is the intuition that computing a general functionality f, that mixes the inputs of the parties, is a harder task than computing the VSS (sharing phase) functionality whose input is taken from a *single* party. Following this (somewhat vague) intuition, we expect to pay more, in terms of round complexity, for f than for VSS. Since the cost of perfectly-secure VSS with optimal resiliency is 3 rounds [27], such a reasoning should yield a lower-bound of 4 rounds. A natural way to formalize the above intuition is to show that a k-round protocol for f can be turned into a $(k-1)$-round VSS protocol. We follow this route and present a novel "protocol-chopping" technique. Details follow.

Chopping a protocol: Consider a k-round protocol π for some functionality f whose properties will be specified later. In such a protocol, each party P_i starts with an input x_i and at the end receives an output y_i. Using standard reductions, we may assume, without loss of generality, that only the first round messages make use of the private point-to-point channels, and all other messages are sent over the broadcast channel. Let us run the protocol π and halt the execution after $k-1$ rounds. At this point, each party holds the public broadcast values that were transmitted in the first $k-1$ rounds, together a private view v_i, that consists of the

party's private input/randomness and the private incoming messages received at the first round. We collect all this information in a vector T, and analyze the properties of T that are induced by the security of π. For example, the privacy of the protocol clearly implies that local parts of T that are available to a t-size coalition I of "corrupted" parties, do not reveal any non-trivial information about the inputs of other "uncorrupted" parties. (In fact, each such I induces an equivalence relation over realizable T's.) More importantly, the fact that the protocol is secure against an active adversary implies that any k-th round extension of T (i.e., a vector of broadcast values) cannot violate correctness even if an adversary controls an I-subset of the parties. We will use these properties (and others) to argue that, when f is chosen properly, the chopped-down protocol, π', realizes VSS.

Extracting a VSS: Roughly speaking, we let one of the parties D play the role of the dealer, and think of her f-input as the secret s. The other parties will choose their f-inputs uniformly at random. The definition of f will guarantee that, under this choice of inputs, the input/output values of any I-coalition (that does not contain the dealer) perfectly hide the value of s. The chopped-down protocol π' will be used as a sharing protocol where v_i (together with the broadcast values from the first $k-1$ rounds) plays the role of the i-th share of party P_i. At the reconstruction phase, each party broadcasts its share v_i, and the parties essentially emulate the last round of π. Views which are clearly corrupted (e.g., inconsistent with too many other views) are excluded, and the functionality f is defined with sufficient redundancy, so that, even in this case, the secret can still be recovered based on the f-inputs/outputs of sufficiently many parties (and even at the presence of corruptions). We further show that an undetected cheating corresponds to an adversarial behaviour in π and is therefore protected by the security properties of the protocol. As one may expect, the most challenging part is to show that after the sharing phase, the dealer is committed to his input. Roughly, we show that a violation of the commitment property allows the dealer to break the "independence of inputs" property in π and effectively correlates her input with the input of an honest party. (Interestingly, this part strongly relies on the security of π against a rushing adversary.)

The actual implementation of these ideas, including the exact definition of the functionality f, turns to be quite subtle, and the reader is referred to the full version of this paper [7] for full details. As part of the proof, we provide general tools for analyzing the state T of chopped-down protocols. We believe that these tools and, more generally, our chopping technique, may lead to new lower-bounds in other contexts.

B. Computing Degree-2 Functionalities in Four-Rounds

Consider a degree-2 functionality f over some finite field $\mathbb{F}$ of size $|\mathbb{F}| \geq n+1$. For simplicity, assume that the functionality f takes an input $x \in \mathbb{F}$ from P_1, and an input $y \in \mathbb{F}$ from P_2, and delivers the product xy to all the parties.[4]

The classical protocols: The standard approach for computing such a functionality will be roughly as follows. At the sharing phase, P_1 and P_2 secret-share their inputs via VSS. At the end of this phase, party P_i holds the shares $X(i)$ and $Y(i)$ where X and Y are degree-t polynomials whose free coefficients are x and y, respectively. Next, party P_i computes the product, $X(i) \cdot Y(i)$, of its shares which lie on the degree $2t$ polynomial $Z = X \cdot Y$. The degree of this polynomial is too high to allow noisy-interpolation (at the presence of $t = \lceil n/3 \rceil - 1$ corrupted points), and therefore the parties apply a degree-reduction sub-protocol which distributively transforms the shares $Z(i)$ into new shares $\hat{Z}(i)$ that lie over a random degree-t polynomial $\hat{Z}$ whose free coefficient equals to $Z(0)$. The latter step can be reduced to (many parallel calls of) VSS. Finally, the parties reveal their $\hat{Z}$-shares. Using noisy interpolation, one can recover the free coefficient $\hat{Z}(0)$ even if t of the shares are corrupted. (Such a noisy interpolation is possible whenever the number of "honest points", $(n-t)$, is two times larger than the number t of corrupted points.)

Letting $R_{\text{vss}} = 3$ denote the round complexity of VSS, we derive a protocol whose round complexity is $2 \cdot R_{\text{vss}} + 1 = 7$. One may try to start the degree-reduction phase earlier, before the sharing phase terminates, however, some of the honest-party shares become only available at the last round of the VSS, and so it is not clear how to carry-out such an optimization.[5]

A more liberal coding scheme: We take a different route and deviate from the above blueprint. Let us start by relaxing the role of degree-reduction. Recall that standard VSS generates, as a by-product, second-level shares. That is, when x is shared, the i-th party gets a first-level share $X(i)$ and, in addition, gets, for every $j \in [n]$, a share $X(j,i)$ of the j-th first-level share $X(j)$ where $X(j,1), \ldots, X(j,n)$ all lie on a degree-t polynomial. Our first insight is that instead of reducing the degree of the product polynomial Z, it suffices to reduce the degrees of the second-level polynomials. That is, while the product polynomial Z remains a degree-$2t$ (re-randomized) polynomial that holds xy as its free-coefficient, our degree-reduction protocol will generate a degree-t second-level sharing for each $Z(j)$. This means that the shares, $Z(j,1), \ldots, Z(j,n)$, should lie on a degree-t polynomial. Moreover, we require the existence of a public

[4] This example is somewhat simpler than the actual complete degree-2 functionalities, and it is chosen for ease of presentation.

[5] One could try to use pre-computed secret-shared multiplicative triples [10], however, in order to generate such triples we need a protocol for degree-2 functionalities.

list G of at least $n-t$ "good" parties, such that for every $P_j \in G$ the second-level shares $Z(j,1),\ldots,Z(j,n)$ are at most t-noisy. Once we have such $n-t$ "good" points we can recover the degree-$2t$ polynomial Z using standard interpolation (which is possible since $n-t>2t$.)[6]

Of course, one should still implement this second-level degree-reduction, and it is not clear why this task is easier than the original one. In short, the difference is that the second-level degree-reduction for a first-level share $Z(j)$ will be lead by the j-th party P_j, and, unlike first-level degree-reduction, our correctness requirements here are relatively modest. When a corrupted P_j misbehaves, we do not care whether the process succeeds as long as this misbehavior is publicly detected. In such a case, we simply remove P_j from the set G. In addition, in this sub-protocol no privacy requirement is needed against P_j, simply because this party "knows" everything (i.e., the polynomial $Z(j)$ is fully known to P_j.)

Second-level degree-reduction: We briefly explain how to implement the second-level degree reduction. Let us assume for now that each party P_j shared, in some preprocessing phase, a triple of degree t polynomials A,B and C with $A(0)\cdot B(0)=C(0)$ such that each party P_i holds $A(i),B(i)$ and $C(i)$. (The round-complexity of the preprocessing phase is ignored for now.) Beaver's well known reduction [10] uses such a multiplicative-triple to obtain a single-round degree-reduction (i.e., to transform shares of degree-t polynomials F and F' into degree-t shares of $F(0)\cdot F'(0)$) that involves a couple of reconstructions.

We get rid of the preprocessing assumption by presenting a VSS-based *centralized* triple-sharing protocol in which a *single* dealer chooses the polynomials A,B and C. (The fact that the protocol is centralized saves the need for distributed degree-reduction.) The protocol requires 4 rounds where the sharing is completed in the first 3 rounds and the verification of the product relation of the secrets is completed in Round 4.

Earlier guided degree-reduction: At this point, we still face one final problem. The inputs to Beaver-based round reduction, namely the second-level sharing of $X(i)$ and $Y(i)$ are ready only at the end of Round 3 (upon conclusion of VSS instances). Therefore, one has to spend an additional 4-th round to complete the degree-reduction (even if the random multiplication triples were prepared in an offline phase). This leads to a 5-round protocol. To resolve this issue, let us (naively) assume for now that P_i receives the second-level polynomials $X(i,\cdot)$ and $Y(i,\cdot)$ already at the end of Round 2. Since P_i chose the random multiplication polynomials A,B,C by herself, this tuple is also ready at this point. As a result, P_i has enough information to help the parties execute the additional round needed for degree-reduction already at Round 3.

Of course, a corrupt P_i can cheat and mislead the computation leading to a faulty execution of Beaver's trick. To cope with this, we let each party verify, at Round 4, whether P_i's guided degree-reduction is consistent with the actual second-level shares that were finalized in Round 3. If a misbehavior is detected, P_i is excluded from the set G. Yet again, having at least $2t+1$ honest parties ensures that we will have enough values on the main $2t$-degree polynomial Z even after excluding the outcomes of Beaver's trick corresponding to all corrupt P_is.

VSS in 2.5 rounds: To complete the description, we still have to come-up with a 3-round VSS in which the second-level polynomials $X(i)$ and $Y(i)$ are available to P_i already after two rounds. While we do not know how to satisfy this requirement, we show how to tweak the 3-round VSS of [34] in a way that guarantees a slightly weaker property: At the end of Round 2, each party P_i holds some preliminary version $X'(i,\cdot)$ and $Y'(i,\cdot)$ of the second-level polynomials $X(i,\cdot)$ and $Y(i,\cdot)$ with the guarantee that for any honest party P_i, the preliminary polynomials, $X'(i,\cdot)$ and $Y'(i,\cdot)$, either fully agree with the final polynomials, or agree with the final polynomials on some universal set of $t+1$ points that will become public later (in the end of round 3). This property still suffices for the final reconstruction. As a side note, we present a simpler recipe for VSS through a new building block, Weak Commitment Scheme, that offers a relatively simpler instantiation compared to the traditional building block Weak Secret Sharing.

C. Completeness of Degree-2 Functionalities over Large Fields

As already mentioned, our starting point is the recent theorem of [4], [5] that establishes the completeness of degree-2 functionalities over the binary field. Their proof proceeds in two steps: (1) Convert the target functionality f into a more friendly functionality F whose circuit is based on a protocol for computing f with the desired security guarantees; (2) Use an "optimized version" of the perfectly-secure garbled circuit (GC) of [31] in order to reduce F into a degree-2 functionality g. Recall that the standard (non-optimized) GC-based reduction yields a randomized functionality that has degree-1 in the inputs and degree-2 in the randomness, leading to an overall degree of 3. In the optimized version, instead of *jointly sampling* each internal random-bit that is employed by the GC, we carefully *partition* the random bits between the parties and grant each party full control on his part of the randomness. Consequently, this randomness can be locally preprocessed in a way that simplifies the residual computation into a degree-2 computation. In general, one cannot just partition the randomness between the parties without violating security, however, [4], [5] show that such

[6] Using the terminology of error-correcting codes, we essentially replace the standard degree-t Reed-Solomon code, $\mathrm{RS}(n,t)$, that tolerates at most $(n-t)/2$ errors by the tensor code $\mathrm{RS}(n,2t)\bigotimes\mathrm{RS}(n,t)$ that tolerates full erasures of at most t columns together with errors in at most $(n-t)/2$ locations in every un-erased column.

a partitioning can be safely applied since F itself is based on a secure protocol for f. In order to handle the active setting, it is shown that any adversarial deviation at the local-preprocessing stage can be mapped into a cheating strategy against the original protocol F, and so such a behavior can be simulated based on the simulator of F.

Our goal is to obtain a large-field version of this result. For this purpose, one may try to employ arithmetic variants of the perfectly-secure GC constructions. While such variants exist (see [3], [6]), they do not seem to allow for degree-2 reduction. (A straightforward adaptation of the local pre-processing technique to the arithmetic setting either violates security or leads to large degree of $|\mathbb{F}| > 2$.) Another option is try to embed the binary degree-2 functionality into a degree-2 functionality over a larger field. As explained in the introduction, while this is trivial in the passive setting, when the adversary honestly follows the protocol, this transformation fails at the presence of an active adversary that may use non-binary inputs. Hence, one has to add a mechanism that forces binary behavior. Designing such a *non-interactive degree-2* mechanism is a challenging task, especially in our perfect error-free setting. Indeed, our mechanism should not only *detect* non-binary behavior, it should also *correct* the output delivered to honest parties (i.e., deliver a "good" output to the honest parties which is consistent with some binary input for the corrupted parties), and *erase* the output that corresponds to non-binary inputs (so that corrupted parties do not learn an output that is induced by a non-binary input).[7] We do not know how to directly obtain such a mechanism for a general functionality.

Fortunately, the complete $\mathbb{F}_2$-quadratic functionality turns to satisfy several "nice" properties that significantly simplify our task. In short, we make two main observations: (1) We can successfully cope ("correct and erase") with a non-binary x when x is a "public" variable that will be revealed to everyone by the functionality; (2) Loosely speaking, the inputs to the Boolean perfectly-secure GC functionality are either "arithmetic-friendly" or "publicly-available". Let us elaborate on these two points.

Coping with public inputs: Say that the input x is a public variable whose value will be known to everyone. In this case, we can construct an ifBin gadget that releases a key B (for *binary*) if and only if $x \in \{0,1\}$. Similarly, we can construct an ifnotBin gadget that releases a key A (for *arithmetic*) if and only if $x \notin \{0,1\}$. Moreover, both gadgets can be implemented by quadratic functionalities that output a tuple of elements:

$$\mathsf{ifBin}(B, x; R_1, R_2) := \big(x \cdot R_1 + B, (1-x) \cdot R_2 + B\big),$$
$$\mathsf{ifnotBin}(A, x; R_1, R_2) := \big(x \cdot R_1, (1-x) \cdot R_2, R_1 + R_2 + A\big),$$

where the "keys" A and B are field elements and R_1 and R_2 are fresh random field elements. Indeed, if x equals to zero or one, the first or second output of ifBin reveals B, whereas for any other field element $x \notin \{0,1\}$ both outputs of ifBin reveal no information about B since the random elements $x \cdot R_1$ and $(1-x) \cdot R_2$ act as one-time pads. (An analysis of similar flavor can be applied to ifnotBin as well.) By using these gadgets one can achieve in principle both correction and erasure. For example, if $F(x,y)$ outputs $(x, g(x,y))$ for some degree-2 function g, we can replace it with the arithmetic functionality that outputs the tuple $(x, g(x,y) + B, g(0,y) + A)$ together with $\mathsf{ifBin}(B, x; R_1, R_2)$ and $\mathsf{ifnotBin}(A, x; R_1, R_2)$. When a non-binary input x is being used, the information $g(x,y)$ is being erased (since the key B is not released) and the alternative "corrected" output $g(0,y)$ can be computed (since the key A is being released).

Arithmetizing the Boolean GC functionality: The Boolean GC functionality employs two types of random inputs: (a) *keys* that are used for encrypted gate-tables; and (b) *wire masks* that are used for masking the value (aka signal) that propagates over each wire. The former are being used only in degree-1 computations either as the keys to one-time pads or as the content that is being encrypted. Correspondingly, there is no harm in taking these values to be general field elements. The wire masks are in turn more sensitive. Each such mask is locally-manipulated via degree 2 computation by some party who "owns" the wire. More importantly, the results of these preprocessed values is being used as a "selector" s for one of two keys via an expression of the form $sK_1 + (1-s)K_0$. Replacing such a selector with a general non-binary field element is problematic both for privacy and for correctness. (Roughly, arithmetic selectors allow to run the circuit over a linear combination of the inputs.) We therefore have to force these values to be binary.

Wire masks must be private, and so we cannot handle them via the ifBin and ifnotBin gadgets. The crucial observation is that the *masked signal* of a wire, which is simply the sum of the wire's mask and signal, is actually public. Furthermore, one can tweak the GC construction so that instead of enforcing binary behavior over masks, it suffices to enforce such a behavior on the masked signals. Following this approach, we integrate the gadgets into the garbled table of each gate, while making sure that a malicious non-binary execution leads to an effective choice of zeroes.

Several technicalities still arise, especially for gates that correspond to broadcast in the original protocol F. In particular, in some cases the adversary may apply a non-binary mask to values that are "owned" by different honest parties. The use of binary extension field guarantees that

[7] Indeed, one can easily notify the parties whether an input x is binary or not, by appending the quadratic "flag" $y = x^2 - x$ to the output of the function. However, "correction" and "erasure" seem to require a degree-2 computation in y which increases the overall degree to 3 or more.

such a non-binary mask will unanimously throw all these binary values outside $\{0, 1\}$, and is therefore translated to a single binary broadcast message. The reader is referred to the full version of this paper [7] for full details.

REFERENCES

[1] Prabhanjan Ananth, Arka Rai Choudhuri, Aarushi Goel, and Abhishek Jain. Round-optimal secure multiparty computation with honest majority. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, pages 395–424, 2018.

[2] Prabhanjan Ananth, Arka Rai Choudhuri, Aarushi Goel, and Abhishek Jain. Two round information-theoretic MPC with malicious security. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, pages 532–561, 2019.

[3] Benny Applebaum. Garbled circuits as randomized encodings of functions: a primer. In *Tutorials on the Foundations of Cryptography.*, pages 1–44. 2017.

[4] Benny Applebaum, Zvika Brakerski, and Rotem Tsabary. Perfect secure computation in two rounds. In *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part I*, pages 152–174, 2018.

[5] Benny Applebaum, Zvika Brakerski, and Rotem Tsabary. Degree 2 is complete for the round-complexity of malicious MPC. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, pages 504–531, 2019.

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

[7] Benny Applebaum, Eliran Kachlon, and Arpita Patra. The round complexity of perfect MPC with active security and optimal resiliency. *Electronic Colloquium on Computational Complexity (ECCC)*, 27:76, 2020. Full version of this paper.

[8] Michael Backes, Aniket Kate, and Arpita Patra. Computational verifiable secret sharing revisited. In *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, pages 590–609, 2011.

[9] Judit Bar-Ilan and Donald Beaver. Non-cryptographic fault-tolerant computing in constant number of rounds of interaction. In *Proceedings of the Eighth Annual ACM Symposium on Principles of Distributed Computing, Edmonton, Alberta, Canada, August 14-16, 1989*, pages 201–209, 1989.

[10] D. Beaver. Efficient Multiparty Protocols Using Circuit Randomization. In J. Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15*, volume 576 of *Lecture Notes in Computer Science*, pages 420–432. Springer Verlag, 1991.

[11] Donald Beaver, Joan Feigenbaum, Joe Kilian, and Phillip Rogaway. Security with low communication overhead. In *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, pages 62–76, 1990.

[12] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 503–513, 1990.

[13] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 1–10, 1988.

[14] Fabrice Benhamouda and Huijia Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, pages 500–532, 2018.

[15] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 136–145, 2001.

[16] Ran Canetti, Joe Kilian, Erez Petrank, and Alon Rosen. Black-box concurrent zero-knowledge requires omega˜(log n) rounds. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 570–579, 2001.

[17] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 11–19, 1988.

[18] Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 383–395, 1985.

[19] Richard Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 364–369, 1986.

[20] Ronald Cramer and Ivan Damgård. Secure distributed linear algebra in a constant number of rounds. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 119–136, 2001.

[21] Ivan Damgård and Yuval Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, pages 378–394, 2005.

[22] Danny Dolev and Rüdiger Reischuk. Bounds on information exchange for byzantine agreement. *J. ACM*, 32(1):191–204, 1985.

[23] Paul Feldman and Silvio Micali. Byzantine agreement in constant expected time (and trusting no one). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 267–276, 1985.

[24] Matthias Fitzi, Juan A. Garay, Shyamnath Gollakota, C. Pandu Rangan, and K. Srinathan. Round-optimal and efficient verifiable secret sharing. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, pages 329–342, 2006.

[25] Sanjam Garg, Yuval Ishai, and Akshayaram Srinivasan. Two-round MPC: information-theoretic and black-box. In *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part I*, pages 123–151, 2018.

[26] Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, pages 468–499, 2018.

[27] Rosario Gennaro, Yuval Ishai, Eyal Kushilevitz, and Tal Rabin. The round complexity of verifiable secret sharing and secure multicast. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 580–589, 2001.

[28] Rosario Gennaro, Yuval Ishai, Eyal Kushilevitz, and Tal Rabin. On 2-round secure multiparty computation. In *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, pages 178–193, 2002.

[29] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM J. Comput.*, 25(1):169–192, 1996.

[30] Yuval Ishai, Ranjit Kumaresan, Eyal Kushilevitz, and Anat Paskin-Cherniavsky. Secure computation with minimal interaction, revisited. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 359–378, 2015.

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

[32] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, pages 244–256, 2002.

[33] Yuval Ishai, Eyal Kushilevitz, and Anat Paskin. Secure multiparty computation with minimal interaction. In *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, pages 577–594, 2010.

[34] Jonathan Katz, Chiu-Yuen Koo, and Ranjit Kumaresan. Improving the round complexity of VSS in point-to-point networks. *Inf. Comput.*, 207(8):889–899, 2009.

[35] Ranjit Kumaresan, Arpita Patra, and C. Pandu Rangan. The round complexity of verifiable secret sharing: The statistical case. In *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, pages 431–447, 2010.

[36] Eyal Kushilevitz, Yehuda Lindell, and Tal Rabin. Information-theoretically secure protocols and security under composition. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 109–118, 2006.

[37] Leslie Lamport and Michael Fischer. Byzantine generals and transaction commit protocols. Technical report, Technical Report 62, SRI International, 1982.

[38] Tal Moran, Moni Naor, and Gil Segev. An optimally fair coin toss. *J. Cryptology*, 29(3):491–513, 2016.

[39] Arpita Patra, Ashish Choudhary, Tal Rabin, and C. Pandu Rangan. The round complexity of verifiable secret sharing revisited. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, pages 487–504, 2009.

[40] Arpita Patra and Divya Ravi. On the exact round complexity of secure three-party computation. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, pages 425–458, 2018.

[41] Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.

[42] Tomas Sander, Adam L. Young, and Moti Yung. Non-interactive cryptocomputing for NC1. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 554–567, 1999.

[43] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[44] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167, 1986.