Zero-Knowledge from Secure Multiparty Computation[*]

Yuval Ishai[†]
Computer Science Dept.
Technion, Haifa, Israel
yuvali@cs.technion.ac.il

Eyal Kushilevitz[‡]
Computer Science Dept.
Technion, Haifa, Israel
eyalk@cs.technion.ac.il

Rafail Ostrovsky[§]
CS and Math Dept.
UCLA, Los Angeles, CA
rafail@cs.ucla.edu

Amit Sahai[¶]
Computer Science Dept.
UCLA, Los Angeles, CA
sahai@cs.ucla.edu

ABSTRACT

We present a general construction of a zero-knowledge proof for an NP relation $R(x, w)$ which only makes a *black-box* use of a secure protocol for a related *multi-party* functionality f. The latter protocol is only required to be secure against a small number of "honest but curious" players.

As an application, we can translate previous results on the efficiency of secure multiparty computation to the domain of zero-knowledge, improving over previous constructions of efficient zero-knowledge protocols. In particular, if verifying R on a witness of length m can be done by a circuit C of size s, and assuming one-way functions exist, we get the following types of zero-knowledge proof protocols:

- **Approaching the witness length.** If C has constant depth over $\wedge, \vee, \oplus, \neg$ gates of unbounded fan-in, we get a zero-knowledge protocol with communication complexity $m \cdot \text{poly}(k) \cdot \text{polylog}(s)$, where k is a security parameter. Such a protocol can be implemented in either the standard interactive model or, following a trusted setup, in a non-interactive model.

- **"Constant-rate" zero-knowledge.** For an *arbitrary* circuit C of size s and a bounded fan-in, we get a zero-knowledge protocol with communication complexity $O(s) + \text{poly}(k)$. Thus, for large circuits, the ratio between the communication complexity and the circuit size approaches a constant. This improves over the $O(ks)$ complexity of the best previous protocols.

Categories and Subject Descriptors: F.2 [Theory of Computation]: Analysis of Algorithms and Problem Complexity

General Terms: Theory, Security.

Keywords: Cryptography, zero-knowledge, secure computation, black-box reductions

1. INTRODUCTION

In this work we establish a new general connection between two of the most fundamental tasks in cryptography: zero-knowledge proofs and secure multi-party computation. Before explaining and motivating this connection, we give some relevant background to put it in context.

A *zero-knowledge proof* protocol [19] allows a prover to convince a verifier in the validity of a statement without revealing any further information about the proof beyond the fact that the statement is true. A more general problem is that of *secure multi-party computation* (MPC) [46, 22, 3, 8]. An MPC protocol allows n players to compute a function of their inputs (also referred to as an "n-party functionality") while maintaining the privacy of their inputs and the correctness of the output. A bit more precisely, the protocol should prevent an adversary which may corrupt at most t players from achieving more than it could have achieved by attacking an idealized protocol in which the function is computed by an external trusted party. In particular, corrupted players should not learn further information about the remaining inputs beyond the output of the function. Zero-knowledge protocols can be viewed as a special case of secure two-party computation, where the function verifies the validity of a witness held by the prover.

Honest vs. dishonest majority. MPC protocols can be divided into two categories: ones that rely on the existence of an honest majority (namely, assume that $t < n/2$) [3, 8, 41] and ones that can offer security even when there is no honest majority [46, 22, 17]. In the case of an honest majority, it is possible to obtain "information-theoretic" security that holds unconditionally, whereas in the case of no honest

[*]Work done in part while the authors were visiting IPAM.

[†]Research supported by grant 1310/06 from the Israel Science Foundation and grant 2004361 from the U.S.-Israel Binational Science Foundation.

[‡]Research supported by grant 1310/06 from the Israel Science Foundation and grant 2002354 from the U.S.-Israel Binational Science Foundation.

[§]Research supported by grant 2002354 from the U.S.-Israel Binational Science Foundation, IBM Faculty Award, Xerox Innovation Group Award, NSF Cybertrust grant no. 0430254, and U.C. MICRO grant.

[¶]Research supported by grant 2004361 from the U.S.-Israel Binational Science Foundation, NSF Cybertrust grants, a Cyber-TA grant, an equipment grant from Intel, and an Alfred P. Sloan Research Fellowship.

majority one needs to settle for computational security that holds under cryptographic assumptions. On the other hand, the honest majority assumption is often too strong, and is meaningless in the important two-party case.

Constructions of these two types of protocols differ vastly in the types of techniques they employ. While protocols from the second category (most notably, zero-knowledge protocols) have occasionally proved useful for the design of protocols from the first category, we are not aware of applications in the other direction. This work is motivated in part by the hope to leverage the efficiency and simplicity of MPC with honest majority for the design of efficient protocols in the case of no honest majority, and in particular for the design of efficient zero-knowledge protocols.

Semi-honest vs. malicious players. Another major distinction is between MPC protocols that offer security against *semi-honest* (aka "honest but curious") players, who follow the protocol's instructions but may try to gain additional information from what they see, and protocols that offer security against *malicious* players, who may arbitrarily deviate from the protocol's instructions in order to compromise the privacy of the inputs or the correctness of the output. We refer to these two types of protocols as being secure in the semi-honest model and the malicious model, respectively. Security in the semi-honest model is typically much easier to realize than security in the malicious model. Remarkably, the celebrated result of Goldreich, Micali, and Wigderson [21, 22] shows how to compile an *arbitrary* MPC protocol Π which securely computes a function f in the semi-honest model into a protocol Π' which securely computes f in the malicious model. The high level technique, known as the "GMW paradigm," is to require players to validate every message they send by supplying a zero-knowledge proof that the message is consistent with the protocol's specification. To implement the zero-knowledge proofs, the compiler needs to look into the code[1] of Π rather than use it as a black box, and the resulting protocol Π' needs to perform multiple cryptographic operations for every gate in the circuits implementing Π. Thus, applying the GMW paradigm may involve a considerable efficiency overhead, which gets worse with the computational complexity of Π. A second source of motivation for our work is the goal of finding alternative "black-box" approaches for boosting security in the semi-honest model into security in the malicious model.

1.1 Our Contribution

We present a general construction of a zero-knowledge proof system Π_R for an NP relation $R(x, w)$ which makes a *black-box* use of an MPC protocol Π_f for a related multi-party functionality f, along with a bit commitment protocol. The functionality f can be efficiently defined by only making a black-box (oracle) access to R. The MPC protocol Π_f may involve an arbitrary number of players n, and only needs to be secure against two *semi-honest* players. In particular, when $n \geq 5$ the MPC protocol only needs to be secure against an honest majority.

The basic variant of our construction proceeds as follows.

[1]When we talk of "black-box" use of the protocol Π, we mean simply invoking the next-message-functions of each player in the protocol without looking inside the details of the circuits or TM's that describe these functions. This is in keeping with the standard usage of the term "black-box" when talking of reductions between cryptographic primitives (cf. [43]).

Define $f(x, w_1, w_2, \ldots, w_n) = R(x, w_1 \oplus w_2 \oplus \cdots \oplus w_n)$. The function f is viewed as an n-party functionality, where x (an NP statement) is known to all n players, and w_i (the i-th share of the witness) is a private input of Player i. Note that f is efficiently defined using an oracle to R. The zero-knowledge protocol Π_R begins with the prover secret-sharing (on her private work-tape) the witness w into n additive shares, picking random $w_1, \ldots, w_n$ such that $w = w_1 \oplus \cdots \oplus w_n$. The prover then "runs in her head" the given n-party protocol Π_f for the functionality f, using the statement x and the shares $w_1, \ldots, w_n$ as inputs for the n players. After this execution is completed, the prover begins her interaction with the verifier. The prover commits to the views of the n players, and the verifier picks a random pair of distinct players i, j and challenges the prover to open the committed views of these players. Finally, the verifier accepts if the opened views are consistent with each other (and with Π_f) and the outputs in these views are 1.

The zero-knowledge property of the protocol follows from the security of Π_f against two semi-honest players. Assuming that Π_f is perfectly correct, violating the soundness requires a cheating prover to generate at least one pair of inconsistent views, which is detected with probability at least $1/\binom{n}{2}$. Using $O(kn^2)$ repetitions, the soundness error can be decreased to 2^{-k}.

The black-box nature of our transformation suggests the possibility of significantly greater efficiency. Towards obtaining more efficient variants of the construction which avoid the cost of soundness amplification via repetition, we employ MPC protocols that tolerate a larger number of corruptions and allow the verifier to open many views at once. In this case, security of Π_f against 2 semi-honest players does not suffice, and we need to rely on MPC protocols that have some form of robustness against t malicious players (where typically $t = \Theta(k)$ and $n = \Theta(t)$). The intuition behind this change is that it will allow the verifier to obtain t views, rather than just two views, from a single execution of the protocol. Security against t *malicious* players guarantees that in order to violate the correctness of the MPC protocol (or the soundness of the zero-knowledge protocol) the inconsistencies between the views must be "well-spread" in a way that opening t random views reveals an inconsistency with overwhelming probability. Note that security against t semi-honest players does not suffice here. Indeed, in this case a single malicious player can cause all other players to have an incorrect output; if this particular player is not picked by the verifier, no inconsistency is revealed and still the output may be incorrect.

Another extension of the basic construction that is needed for our applications is to the case where Π_f produces an incorrect output with negligible probability. This case makes even the simple basic construction fail, as it allows the adversary to cheat by biasing the randomness used for generating the views (but otherwise behaving according to the protocol). We show how to overcome this obstacle using standard cryptographic techniques.

Applications. To demonstrate the usefulness of our general approach, we translate previous results on the efficiency of MPC to the domain of zero-knowledge, improving over some previous constructions of efficient zero-knowledge protocols. In particular, if verifying $R(x, \cdot)$ on a witness w of length m can be done by a circuit C of size s, we get the following types of zero-knowledge proof protocols.

Simple zero-knowledge. By plugging in standard MPC protocols for the semi-honest model, such as the 2-private 5-player BGW protocol [3] or the 2-private 3-player GMW protocol [22, 17], we get simple zero-knowledge protocols with complexity $O(ks)$. (Here and in the following k denotes a security parameter, and the soundness error is at most 2^{-k}.) These protocols provide more efficient alternatives to the well-known zero-knowledge protocols based on Hamiltonicity or 3-Colorability.

Approaching the witness length. If C has constant depth over $\wedge, \vee, \oplus, \neg$ gates of unbounded fan-in, we get a zero-knowledge proof protocol with $m \cdot \text{poly}(k) \cdot \text{polylog}(s)$ bits of communication. This protocol can be implemented using a one-way function.[2] Alternatively, our technique yields a zero-knowledge proof system in a non-interactive model with preprocessing. The complexity of our protocols is comparable to that of a previous protocol of Kalai and Raz [28]. The latter, however, only yields zero-knowledge *argument* systems and relies on stronger assumptions, but requires less communication for implementing the setup. The above zero-knowledge protocols rely on the MPC protocol from [1] (which in turn relies on techniques of [42, 45]). We note that the protocols we obtain are essentially the best one could hope for both in terms of the underlying assumption (one-way functions) and communication complexity. Indeed, a lower communication complexity for zero knowledge *proofs* (as opposed to arguments [31]) would imply surprisingly efficient probabilistic algorithms for SAT [20].

"Constant-rate" zero-knowledge. Assuming that one-way functions exist, we get a zero-knowledge proof protocol for an *arbitrary* circuit C of size s and bounded fan-in with communication complexity $O(s) + \text{poly}(k)$. Thus, for large circuits, the ratio between the communication complexity and the circuit size approaches a constant. This improves over the $O(ks)$ complexity of the best previous protocols (e.g. [10]). Our zero-knowledge protocol relies on an MPC protocol from [13], optimized to work over constant-size fields using secret sharing based on Algebraic-Geometric codes [9].

Perspective. Our construction has direct relevance to the two motivating goals discussed above. First, it establishes a new general connection between zero-knowledge and MPC, which in particular allows us to exploit information-theoretic MPC protocols, that were designed in the setting of an honest majority, for the design of zero-knowledge proofs. The latter, in turn, can be used for the design of general (computationally secure) MPC protocols in the setting of no honest majority. This new connection also suggests, as the above applications illustrate, that the efficiency of zero-knowledge proofs might be closely related with that of MPC with honest majority. In particular, some remaining open questions concerning the efficiency of zero-knowledge proofs can now be linked with similar open questions in the area of MPC.

Second, our construction shows a useful class of applications for which security in the semi-honest model can be boosted in a *black-box* way into security in the malicious model. It is instructive to note that it is generally impossible to *directly* construct a protocol Π' which securely computes a function R in the malicious model by making a black-box use of a protocol Π which securely computes R in the semi-honest model. For instance, if R is a zero-knowledge functionality, in the semi-honest model we can consider a trivial protocol Π in which the prover simply sends to the verifier the output of R on its input (x, w). Clearly Π is generally useless for the construction of Π'. By modifying R into a related multi-party functionality f we avoid this impossibility.[3]

We finally note that it is possible to extend our results for zero-knowledge proofs to the case of arbitrary two-party functionalities whose output depends on the input of a single player. (These include, for instance, useful "commit and prove" functionalities.) We leave open the possibility of obtaining a similar black-box construction for a larger class of secure computation tasks.

1.2 Related Work

Black box reductions. A rich body of work, initiated by the seminal paper of Impagliazzo and Rudich [26], attempts to draw the lines between possibility and impossibility of black-box reductions in cryptography. In particular, black-box constructions of MPC protocols from various cryptographic primitives have been previously suggested [30, 12, 27]. Our work differs from all this work in that it treats not only the underlying cryptographic primitives as a black box but also the *functionality* that should be realized. In contrast, previous constructions address either the case of some fixed functionality or alternatively make a non-black-box use of the functionality.

Efficiency of zero-knowledge. There has been a vast body of work on improving the efficiency of interactive and non-interactive zero-knowledge proofs; see [5, 10, 32, 33, 11, 6, 23, 28] and references therein. It is important to note in this context that by settling for computational soundness, one can get asymptotically very efficient zero-knowledge *argument* systems with a polylogarithmic communication complexity. These argument systems can be either interactive and based on collision-resistant hash functions [31] or non-interactive in the random oracle model [34]. Our focus on proofs rather than arguments is motivated both from a theoretical and from a practical point of view. From a theoretical point of view, we get a stronger notion of soundness under weaker assumptions. From a more practical point of view, the PCP-based constructions of arguments are still quite far from being efficient in practice. Furthermore, our results are independently motivated by the general goal of boosting security against semi-honest players into security against malicious players in a black-box way. This motivation applies both to proofs and to arguments.

Organization. In Section 2 we give some preliminaries. The basic construction of zero-knowledge protocols from MPC in the semi-honest model is described in Section 3, including (in Section 3.2) the application to zero-knowledge protocols whose complexity is close to the witness length. In Section 4, we present our constructions of zero-knowledge protocols from MPC in the malicious model, including the application to constant-rate zero-knowledge (in Section 4.2). Various technical (but standard) implementation issues are deferred to Appendix A. For lack of space, some of the details are postponed to the full version.

[2] A similar result was obtained independently and using different techniques by Kalai and Raz [29].

[3] Here it is crucial to insist that f make a black-box use of R; otherwise the output of f can encode the description of a circuit computing R, allowing to use the standard (non-black-box) GMW paradigm.

2. PRELIMINARIES

We use standard definitions of our main cryptographic primitives. Below we only highlight the main aspects of these definitions that are important for our purposes.

Zero-knowledge. We use the standard notion of zero-knowledge proofs from the literature [19, 21] (see also [16]), adapted to the case where the honest prover should be efficient. Towards a more refined complexity analysis we will sometimes assume that both the prover and the verifier are additionally given a security parameter 1^k and require the soundness error to be bounded by 2^{-k}. As long as $k = \omega(\log |x|)$ the standard definition is satisfied. For the sake of simplicity we do not explicitly consider the stronger *proof of knowledge* property (cf. [16]). However, our general constructions can be shown to satisfy this property as well.

Idealized primitives. When describing our zero-knowledge protocols, it will be convenient to first describe and analyze them in a *hybrid* model in which idealized versions of primitives such as bit commitment or coin-flipping are available. We then explicitly describe how to instantiate invocations of the ideal primitives in order to get a protocol in the plain model. Such a modular design approach is common in the area of MPC (cf. [7, 17]). However, since we are dealing with *proofs* rather than arguments, we cannot apply off-the-shelf composition theorems to deduce the security of the final protocol, and need to argue each case separately.

Secure multi-party computation (MPC). We use standard definitions of MPC from the literature [7, 16]. Our basic model assumes synchronous communication over secure point-to-point channels. (Some protocols will also rely on a broadcast primitive.)

We let n denote the number of players P_i. A protocol Π is specified via its next message function. That is, $\Pi(i, x, w_i, r_i, (m_1, \ldots, m_j))$ returns the set of n messages (and, possibly, a broadcast message) sent by P_i in Round $j+1$ given the public input x, its local input w_i, its random input r_i, and the messages it *received* in the first j rounds. (In the case of statistical or computational security, Π receives a security parameter k as an additional input.) The output of Π may also indicate that the protocol should terminate, in which case Π returns the local output of P_i. The *view* of P_i, denoted by V_i, includes w_i, r_i and the messages *received* by P_i during the execution of Π. Note that the messages outgoing from P_i as well as its local output can be inferred from V_i and x by invoking Π. We say that a pair of views V_i, V_j are *consistent* (with respect to some public input x) if the outgoing messages implicit in V_i, x are identical to the incoming messages reported in V_j and vise versa.

We consider secure protocols in both the semi-honest and the malicious models. In the semi-honest case, we may break the security requirements into privacy and correctness. We say that Π realizes a deterministic n-party functionality $f(x, w_1, \ldots, w_n)$ with *perfect correctness* if for any execution by honest players (with arbitrary inputs and random inputs r_i) all players output the correct value of f. We say that Π realizes f with *perfect t-privacy* if there is a PPT simulator $\textsc{Sim}$ such that for any inputs $x, w_1, \ldots, w_n$ and every set of corrupted players $T \subseteq [n]$, where $|T| \leq t$, the joint view of players in T is distributed identically to $\textsc{Sim}(T, x, (w_i)_{i \in T}, f_T(x, w_1, \ldots, w_n))$. The relaxations to statistical correctness and statistical or computational privacy are defined in the natural way.

In the malicious model, security cannot be generally broken into privacy and correctness. However, for our purposes we only need the protocols to satisfy a weaker notion of security in the malicious model that is implied by the standard general definition. Specifically, it suffices that Π be correct and private as in the semi-honest model, and moreover it should be perfectly (resp., statistically) *t-robust* in the following sense. For any computationally unbounded adversary corrupting a set T of at most t players, and for any public input x, if there is no $w = (w_1, \ldots, w_n)$ such that $f(x, w) = 1$, then the probability that some uncorrupted player outputs 1 is 0 (resp., is at most 2^{-k}).

Finally, we will also need to distinguish between the case of *adaptive* security, where the adversary is allowed to pick which players to corrupt during the execution of the protocol, and *non-adaptive* security, in which the identity of corrupted players has to be picked in advance. By default, we only assume protocols to be non-adaptively secure.

3. ZK FROM MPC IN THE SEMI-HONEST MODEL

In this section we present our basic constructions of zero-knowledge protocols, which rely on MPC protocols in the semi-honest model.

Let L be a language in NP and let $R(x, w)$ be a corresponding NP-relation. Let f be the following $(n+1)$-argument function ($n \geq 3$), corresponding to R:

$$f(x, w_1, \ldots, w_n) = R(x, w_1 \oplus \ldots \oplus w_n),$$

where $\oplus$ here denotes bitwise exclusive-or of strings (all of the same length). From here on, we will view f as an n-party functionality, where the first argument x is a public input known to all n players, w_i is a private input of player P_i, and the output is received by all players. We will sometimes ignore the public input x, viewing f as an n-argument function specified by x.

Let Π_f be an n-party protocol which realizes f with *perfect* correctness and either perfect, statistical, or computational 2-privacy (in the semi-honest model). We now describe a zero-knowledge protocol Π_R for R. The prover and verifier are both given an input x (an instance of L). The prover is also given a witness w and they both have a black-box access to the MPC protocol Π_f. The zero-knowledge protocol also makes use of a statistically-binding string commitment protocol $\textsc{Commit}$. For simplicity, we assume that $\textsc{Commit}$ is perfectly binding and non-interactive (such a commitment can be based on any one-way permutation, cf. [16]); the protocol and its analysis easily extend to rely on interactive and statistically binding commitments, which can be based on any one-way function [25, 35]. The zero-knowledge protocol Π_R proceeds as follows:

1. The prover picks at random $w_1, \ldots, w_n \in \{0,1\}^m$ whose exclusive-or equals the witness w. She emulates "in her head" the execution of Π_f on input $(x, w_1, \ldots, w_n)$ (this involves choosing randomness for the n players and running the protocol). Based on this execution, the prover prepares the views $V_1, \ldots, V_n$ of the n players and separately commits to each of these views by sending $\textsc{Commit}(V_1), \ldots, \textsc{Commit}(V_n)$ to the verifier.
2. Verifier picks at random distinct player indices $i, j \in [n]$ and sends them to the prover.
3. Prover "opens" the corresponding two views; i.e., she sends $\textsc{Decommit}(V_i), \textsc{Decommit}(V_j)$.

4. Verifier accepts if and only if: (1) the decommitments sent by the prover match the commitments from Step 1, (2) the outputs of both P_i and P_j (which follow from their views) are 1, and (3) the two views are consistent with each other.

Completeness: If $(x, w) \in R$ and the prover is honest then, since $w_1 \oplus \ldots \oplus w_n = w$ and Π_f is perfectly correct, the views $V_1, \ldots, V_n$ always have output 1. Since these are valid views of the protocol, they are always consistent with each other.

Soundness: If $x \notin L$ this means that $R(x, w) = 0$ for all w. Hence, by the perfect correctness of Π_f, for all choices of $w_1, \ldots, w_n$ and all choices of randomness for the players, the output of all n players must be 0. Therefore, considering the n views committed to by the prover in Step 1, either in all views the output is 0, or there exist two views which are inconsistent. In the former case the verifier always rejects, while in the latter case he rejects with probability at least $1/\binom{n}{2}$ (which is his probability to select an inconsistent pair i, j). The error probability can be reduced to 2^{-k} by repeating the protocol $O(kn^2)$ times.

Zero-Knowledge (sketch):[4] The verifier gets to see the views of two players P_i, P_j. These views include the inputs w_i, w_j of these players which are just random m-bit strings, as well as their random inputs and communication which, by the 2-privacy of Π_f, can be simulated based on their input and the output of the protocol (i.e., 1); namely, by executing $\textsc{Sim}((i, j), x, (w_i, w_j), 1)$, where $\textsc{Sim}$ is the simulator guaranteed by the 2-privacy of Π_f.

Theorem 3.1. *If Π_f realizes f with* perfect *correctness and either perfect, statistical, or computational 2-privacy (in the semi-honest model), then Π_R, described above, is a zero-knowledge proof protocol for the NP-relation R. Moreover, Π_R can be implemented using a black-box access to Π_f and to a one-way function.*

Additional remarks:

1. If Π_f is not perfectly correct, a cheating prover may violate the soundness of Π_R: on $x \notin L$, she picks w_i and r_i on which the protocol incorrectly outputs 1, hereby making the verifier incorrectly accept (no matter what indices i, j it chooses). We deal with the issue of imperfect correctness in Section 3.1 below.
2. In the above protocol it suffices to share the witness w among 3 (of the n) players.
3. We will later be interested in minimizing the amount of communication in the zero-knowledge protocols. It is instructive to (briefly) analyze the communication complexity of this basic construction. The communication in Π_R consists mainly of commitments to the views in Π_f. Implementing a commitment to a string of length ℓ costs $\ell + k$ bits of communication, where k is the security parameter. Note that the total length of the views is equal to $O(|w|)$ plus the communication and randomness complexity of Π_f.
4. The description and analysis of Π_R can easily be extended to accommodate protocols Π_f that use more powerful communication channels. For instance, we can assume that each pair of players have oracle access to an arbitrary 2-party functionality, where both players send inputs to and receive outputs from this functionality. The notion of consistent views can naturally be extended for this case; i.e., the verifier can check that the reported outputs from the oracle are consistent with its inputs. (Note that such oracles do not trivialize the design of Π_f because of the 2-privacy requirement.) In particular, we may assume that each pair of players are connected via an *OT channel* [40, 14], in which the sender has two inputs s_0, s_1 and the receiver a selection bit b, and the receiver gets s_b. It is also easy to extend Π_R to the case where Π_f employs broadcast channels, by simply having the prover send to the verifier all broadcast messages.[5]
5. This protocol, as well as most of the following ones, is actually an Arthur-Merlin protocol.

Zero-Knowledge from the BGW or GMW Protocols. Most standard MPC protocols for the semi-honest model can be used in the above transformation. These include the perfectly private $(2, 5)$-BGW protocol [3] as well as the $(2, 3)$-GMW protocol [22, 17] (either with OT channels as above, in which case the protocol is perfectly private and more efficient, or with a computationally private implementation of the OT channels).

3.1 The Case of Statistical Correctness

As noted above, our basic construction relies on the MPC protocol being perfectly correct. We now describe a variant of the basic construction that allows Π_f to be statistically correct. The following modification of Π_R starts with the prover committing to the inputs w_i of the players. Then the prover and the verifier invoke a coin flipping procedure whose final outcome, $r_1, \ldots, r_n$, is known only to the prover. Finally, when revealing two views, the verifier is able to verify that the "correct" random inputs were used. Note that we will invoke an idealized coin flipping oracle in the description below. In an actual implementation, one can use sequential repetition of Blum's protocol [4] (see Appendix A). The modified zero-knowledge protocol Π_R proceeds as follows:

1. The prover picks at random $w_1, \ldots, w_n \in \{0, 1\}^m$ such that $w_1 \oplus \ldots \oplus w_n = w$. She commits to the inputs w_i as well as to n random strings $r_1^P, \ldots, r_n^P$. by sending $\textsc{Commit}(w_1), \ldots, \textsc{Commit}(w_n)$ and $\textsc{Commit}(r_1^P), \ldots, \textsc{Commit}(r_n^P)$.
2. The prover and verifier invoke "coin-flipping" to generate n random strings $r_1^V, \ldots, r_n^V$ known to both. (See Appendix A for implementation details.)
3. The prover emulates "in her head" the execution of Π_f on input $(x, w_1, \ldots, w_n)$, using randomness $r_i = r_i^P \oplus r_i^V$ for each player P_i. Based on this execution, the prover prepares the views $V_1, \ldots, V_n$ of the n players in the protocol and commits to these views by sending $\textsc{Commit}(V_1), \ldots, \textsc{Commit}(V_n)$ to the verifier.

[4]For now, we assume the use of ideal commitment. A sketch of how this proof changes when the ideal commitment is replaced with an actual commitment protocol, in the plain model, is given in Appendix A; note that handling the commitment is standard and follows the work done in the context of the GMW zero-knowledge protocol [21].

[5]This extension is mostly relevant to MPC in the malicious model which is used in Section 4. Interestingly, broadcast in our setting is very cheap, while usually in the context of MPC it is considered an expensive means of communication.

4. Verifier picks at random distinct $i, j \in [n]$ and sends them to the prover.
5. Prover "opens" the corresponding two views and her corresponding shares of the random inputs; that is, she sends DECOMMIT(V_i), DECOMMIT(V_j), DECOMMIT(r_i^P), and DECOMMIT(r_j^P).
6. Verifier accepts if and only if the decommitments sent by the prover match the corresponding commitments, the outputs of both P_i and P_j are 1, the two views are consistent with each other and with the opened w_i, and their random inputs satisfy $r_i = r_i^P \oplus r_i^V$ and $r_j = r_j^P \oplus r_j^V$.

Completeness, as before, is easy to verify. The idea behind the zero-knowledge property is similar to the basic case. The hiding property of the commitment guarantees that the verifier cannot influence the randomness used in Π_f and cannot get more information other than the views of players P_i, P_j. These views, as before, can be simulated using the simulator SIM guaranteed by the 2-privacy of Π_f.

Soundness: Again, we analyze the soundness in a hybrid model. If in the committed views the randomness r_i for each player P_i is indeed $r_i^P \oplus r_i^V$ (as it should be) and if all views are consistent, then the r_i are uniformly distributed and, by the statistical correctness of Π_f, the output of all players is 0 with overwhelming probability. Otherwise, either the randomness used by the protocol (for at least one player) is not as it should be, in which case the verifier rejects with probability at least $2/n$, or there are inconsistent views, in which case the verifier rejects with probability at least $1/\binom{n}{2}$. (As before, $O(nk^2)$ sequential repetitions drive the cheating probability down to 2^{-k}.) Note that the statistical correctness of Π_f is only guaranteed if the random inputs r_i are chosen independently of the inputs x, w_i. Hence, it is necessary that the prover commit to $w_1, \ldots, w_n$ in Step 1.

THEOREM 3.2. *If Π_f realizes f with* statistical *correctness and either perfect, statistical, or computational 2-privacy (in the semi-honest model), then Π_R, described above, is a zero-knowledge proof protocol for the NP-relation R. Moreover, Π_R can be implemented using a black-box access to Π_f and to a one-way function.*

3.2 Application: Approaching the Witness Length

In this section, we present (interactive and non-interactive) zero-knowledge protocols whose communication complexity is $m \cdot \text{poly}(k) \cdot \text{polylog}(s)$, where m is the witness length and s is the size of the witness verification circuit $C_x(w)$ verifying $R(x, \cdot)$. This holds whenever C_x has a constant depth.

The zero knowledge protocols rely on a simple version of an MPC protocol from [1] (the protocol from [1] has stronger properties that we will not need).

FACT 3.3. *[1] Let C be a constant depth circuit (using $\vee, \wedge, \oplus, \neg$ gates with unbounded fan-in) of size s and input w of length m. Then, for $n = \text{polylog}(s)$, and any partition $(w_1, \ldots, w_n)$ of w between n players, there is an n-party MPC protocol Π which computes $C(w_1, \ldots, w_n)$ with statistical correctness and 2-privacy, where the communication complexity and randomness complexity of Π are at most $m \cdot k \cdot \text{polylog}(s)$.*

For the interactive case, all we need to do is to plug the protocol of Fact 3.3 (applied to the constant-depth circuit computing the functionality f obtained from C_x) into the construction of Section 3.1.

The non-interactive case. Next, we turn to the case of non-interactive zero-knowledge with preprocessing, previously considered in [28] and [33]. In the description below, it is assumed that the preprocessing phase is run by a trusted dealer who sends randomized messages to the prover and the verifier (before the input x is available) and then disappears. An implementation of this preprocessing step by an actual protocol between the prover and verifier (using enhanced trapdoor permutations) is given in the full version.

A first idea for obtaining a non-interactive protocol is to replace the (interactive) commit-and-challenge approach by the following implementation of non-interactive 2-out-of-n OT. In the preprocessing step, the dealer sends to the prover n random encryption keys and to the verifier a random subset of two of these keys (whose identity is unknown to the prover). In the online phase, the prover generates n views as in the interactive protocol and sends the encryption of each view V_i using the corresponding key. The verifier learns a random pair of views and, as in the basic protocol, verifies their consistency. (This can be repeated in parallel to amplify soundness.)

The statistical correctness of the underlying MPC protocol Π poses an additional difficulty. Picking the randomness to be used in the protocol Π during the preprocessing step is problematic because the prover may choose for some $x \notin L$ its input w *after* getting to see this randomness. Here, however, we can use a simple modification of [1] that gives a family of protocols Π_ρ, $\rho \in \{0,1\}^{mk \cdot \text{polylog}(s)}$, such that each Π_ρ has roughly the same complexity as Π, and a random protocol from this family is perfectly correct and private except with 2^{-k} probability over the choice of ρ. The protocol can now proceed as in the perfect case, except that a random ρ is given to both the prover and the verifier during the preprocessing step, and ρ is used to define a protocol Π_ρ to be used in the proof. (If the choice of x can depend on ρ, the length of ρ should be increased roughly by $|x|$.) We defer further details on the non-interactive setting to the full version.

4. ZK FROM MPC IN THE MALICIOUS MODEL

In this section, we aim at getting zero-knowledge protocols with negligible soundness error 2^{-k} while avoiding the naive repetition that carries a multiplicative overhead of $\Omega(k)$ in the complexity. We do this by strengthening the security requirement of Π_f to *t-security* in the *malicious* model. (This should be contrasted with *2-privacy* in the *semi-honest* model used in the previous section.) In fact, we will only need the MPC protocols to be t-private in the semi-honest model and (perfectly or statistically) *t-robust* in the malicious model, as defined in Section 2.

We start by assuming that Π_f is perfectly robust. We will pick our parameters such that t, n are both $\Theta(k)$. The zero-knowledge protocol in this case is identical to the protocol in the basic construction except that the verifier opens t randomly selected views and checks for their consistency. (In the actual implementation of commitments, we will also need to add a "preamble" to our protocol in which the verifier commits to its choices of t views using a statistically hiding commitment scheme. Using the very recent result of [24], the latter can be based on any one-way function, which is the minimal assumption required for non-trivial zero-knowledge [38]. Because of statistical hiding, this does not affect the soundness proof sketched below and is only required for proving the zero-knowledge property. For further details, see Appendix A.)

The completeness and zero-knowledge properties in the hybrid model remain as before and only the soundness requires a new proof. The intuition that underlies the soundness argument is as follows. If all inconsistencies can be resolved by changing at most t views, the protocol execution committed to is one that could be realized by an adversary corrupting at most t players. By the t-robustness of Π_f, in such an execution on an input $x \notin L$ all honest players must output 0, an event that will be noticed by the verifier with overwhelming probability. On the other hand, if the inconsistencies are not confined to t views, the t opened views will reveal to the verifier at least one inconsistency with overwhelming probability.

To formally argue the soundness, consider the following *inconsistency graph* G, defined based on the n committed views $V_1, \ldots, V_n$. The graph has n vertices (corresponding to the n views) and there is an edge (i, j) in G if the views V_i, V_j are inconsistent. We analyze the soundness according to two cases, showing that the verifier rejects any $x \notin L$ with overwhelming probability in both cases.

Case 1: There exists in G a vertex cover set B of size at most t. We argue that in this case, the output in all views V_i, for $i \notin B$ must be 0. To see this, we consider an execution of the protocol Π_f where the adversary corrupts the set of players B and behaves in a way that the views of any player P_i, for $i \notin B$, is V_i. By the perfect t-robustness of Π_f, such a set should not influence the output of the honest players which must be 0. Such an execution is obtained by choosing all the messages from $P_i \in B$ to $P_j \notin B$ as in the view V_j. (The intuition here is that we have a "small" vertex cover that is used to "explain" all the inconsistencies among the views.) Finally, if in all views V_i, for $i \notin B$, the output is 0, it suffices that the verifier, among its t choices, will pick at least one player not in B. The probability that this does not happen is at most $(t/n)^t = 2^{-\Omega(n)} = 2^{-\Omega(k)}$.

Case 2: min-VC$(G) > t$. In this case the graph G must have a matching of size $> t/2$ (otherwise, if the maximum matching contains $\leq t/2$ edges, the vertices of this matching form a VC of size $\leq t$). If the verifier picks at least one edge of G it will reject. The probability that the t vertices (views) that the verifier picks miss all the edges of G is smaller than the probability that it misses all edges of the matching which is again at most $2^{-\Omega(n)} = 2^{-\Omega(k)}$. (Note that the advantage of considering a matching is that we get independence between the edges.)

Theorem 4.1. *If Π_f realizes f with perfect t-robustness (in the malicious model) and perfect, statistical, or computational t-privacy (in the semi-honest model) and if $t = \Theta(k)$ and $n = \Theta(t)$, then Π_R, described above, is a zero-knowledge proof protocol for the NP-relation R whose soundness error is $2^{-\Omega(k)}$. Moreover, Π_R can be implemented using a black-box access to Π_f and any one-way function.*

A final change we will need in order to eliminate the $O(k)$ multiplicative overhead of the basic construction is to avoid the additive secret sharing of w. Instead, we will use an MPC protocol Π_f for the following modified functionality f. A special player I (an "input client" in the terminology of [13]) holds the entire witness w, and only a special player O (an "output client") receives the output $R(x, w)$. (The NP statement x, as before, is known to all players.) In addition to I and O there are n servers who have no private inputs and no output. The protocol Π_f is assumed to be secure against a malicious adversary who may corrupt at most t servers. (Each of the clients I and O may be either corrupted or uncorrupted.) This implies, in particular, that: (1) unless O is corrupted the adversary should be unable to make O output 1 if x is a false statement; (2) unless I is corrupted, the adversary should not learn any information about w other than $R(x, w)$. We note that the MPC protocol from [13] on which we will rely is already presented in this client-server framework.

The zero-knowledge protocol in this case is the same as the protocol in the previous construction except that: (1) instead of secret-sharing w between n players, w is directly used by the prover as an input to I; (2) the verifier opens the views of t randomly selected servers along with the view of the output client O and checks for their consistency; furthermore, it checks that O outputs 1. The view of the input client I is never opened, as this would reveal w.

The analysis of the modified construction is the same as before, with the exception that the view of one player, I, is never opened. However, this only has the effect of decreasing the size of the maximum matching by 1, corresponding to missing an edge between I and some server in the original matching. Note that I can only be involved in a single edge of the matching. Thus, as long as $t = \Theta(k)$ the soundness error remains essentially unchanged.

4.1 The Case of Statistical Security

In this section we describe a variant of the above construction that applies to the case where Π_f is only statistically robust. This is motivated by the application to constant-rate zero-knowledge which is sketched in Section 4.2.

If Π_f is only statistically t-robust, a malicious adversary corrupting I and at most t servers can have a negligible probability of cheating, namely making O output 1 when there is no w such that $R(x, w) = 1$. Interestingly, in this case the modification of the basic protocol presented in Section 3.1 does not have the desired level of soundness. This is because the prover, while being committed to random inputs on which it has no control, is allowed to generate the views V_i only *after* it is given the random inputs of *all* players. In such a case a single malicious player may suffice for successfully cheating the honest players. If this particular player is not picked by the verifier, no inconsistency is revealed and still the output may be incorrect.

To get around this difficulty, we need to separate the random inputs r_i that are used for the privacy of the protocol from ones that are used to ensure its robustness. Below we describe a construction that works for protocols that are broken into two phases, Phase I and Phase II, where following Phase I the players obtain a public random string r of length ℓ that is used in Phase II. This can be generalized to protocols that have more than two phases; however, the crucial point is that each string r generated in the middle of the protocol must be unpredictable during previous phases. The robustness of the two-phase protocol should hold also with respect to an *adaptive* adversary that may choose (some of) the t corrupted players after seeing the coins r. More precisely, we require that for any inputs x, w and random inputs r_i, the probability of cheating (i.e., breaking the robustness requirement) by a malicious and possibly adaptive adversary is negligible in k, where the probability is over the choice of r. This strong *adaptive robustness* property, that will be satisfied by the concrete MPC protocols we will use, is crucial for the security reduction to go through.

The zero-knowledge protocol obtained from such a two-phase MPC protocol has the following form. (For the sake of generality, we describe the modified construction in the original n-player model. Its analysis for the client-server model we actually use follows as a special case.)

1. The prover picks at random $w_1, \ldots, w_n \in \{0,1\}^m$ such that $w_1 \oplus \ldots \oplus w_n = w$ and a random input r_i for every server P_i. She emulates the views up to the end of Phase I, denoted by $(V_1^1, \ldots, V_n^1)$, and commits to all of them.
2. The prover and the verifier invoke a coin flipping oracle to generate a string r of length ℓ. (This oracle will be implemented using simulatable coin-flipping; see Appendix A.)
3. The prover continues to run the protocol in her head, using the string r generated in the previous step, and produces a vector $(V_1^2, \ldots, V_n^2)$ of the views of Phase II. The prover commits to these n views.
4. The verifier picks at random distinct $i_1, \ldots, i_t \in [n]$ and sends them to the prover.
5. The prover "opens" the corresponding $2t$ commitments $V_{i_1}^1, \ldots V_{i_t}^1, V_{i_1}^2, \ldots V_{i_t}^2$.
6. The verifier accepts if and only if the opened viewed are all consistent with each other and with r and the output in these views is 1.

The completeness and zero-knowledge properties (in the hybrid model) are as in the basic case. Soundness follows by comparing the adversary's winning probability in the following two games.

ZK soundness game. The adversary chooses a false statement $x \notin L$ and then runs the above protocol with the honest verifier (applying an arbitrary prover strategy). It wins if in the end of Step 3 the inconsistency graph G induced by the full committed views (including both phases) has min-VC$(G) \leq t/2$.

MPC game. The adversary chooses a false input x and arbitrary inputs $w_1, \ldots, w_n$ and random inputs $r_1, \ldots, r_n$ for the n players in the two-phase MPC protocol. It may then adaptively corrupt a total of up to t players during the execution of the protocol. (In particular, some players may be corrupted only after learning r.) The adversary wins if some player that remains uncorrupted at the end of the protocol outputs 1.

CLAIM 4.2. *The prover's winning probability in the ZK soundness game is no bigger than the adversary's winning probability in the MPC game.*

Proof sketch: We show how to simulate an adversary playing the ZK game by an adversary playing the MPC game. The MPC adversary runs the ZK adversary up to the end of Step 1, and extracts the n views of phase I to which it committed. (Such an extraction is possible because adversaries are unbounded.) It finds a set T_1 of size at most $t/2$ which forms a minimal vertex cover in the inconsistency graph induced by these partial views. Now it moves to corrupt Phase I in the MPC protocol. It takes x as generated by the ZK adversary, and the inputs $w_1, \ldots, w_n$ and the random inputs $r_1, \ldots, r_n$ from the extracted views. Now it corrupts the players in T_1, causing them to send messages according to the views. (This is possible because T_1 forms a vertex cover in the inconsistency graph of Phase I.) Now the MPC adversary continues to run the MPC protocol and obtains a random string r which it feeds to the ZK adversary in Step 2. It continues to run the ZK adversary to the end of the ZK protocol, extracting the final set of views in Phase II. Let T_2 be a minimal vertex cover of size at most $t/2$ in the final inconsistency graph. The MPC adversary corrupts all players in T_2 and makes the final views in the MPC protocol identical to the final views in the ZK protocol. $\square$

By the strong adaptive t-robustness of the MPC protocol (which is assumed to hold for an arbitrary choice of the private inputs r_i), the MPC adversary can only win the game with negligible probability. It follows that in order to fool the verifier, the ZK prover should generate an inconsistency graph with a vertex cover of size $> t/2$, in which case it will be caught except with $2^{-\Omega(k)}$ probability.

We note that the factor 2 gap between the size of the vertex cover and the MPC corruption threshold seems inherent to the "on-line" nature of adaptive corruptions. In the end of Phase I, the MPC adversary cannot predict the set that will serve as a minimal vertex cover in the final graph.

4.2 Application: Constant-Rate ZK

In this section we sketch the construction of zero-knowledge protocols whose communication complexity is linear in the circuit size s, by applying the general construction from Section 4.1 on top of a variant of the MPC protocol from [13]. Further details are postponed to the full version.

The protocol from [13] works in the same client-server model we described above. It also conforms to the requirements of a two-phase protocol, as described above, with $t = \Omega(n)$. Thus, the general construction from Section 4.1 can be applied. In fact, Phase II can take the degenerate form of only involving broadcast messages. Thus, in Step 3 of the general construction the prover can send these views to the verifier instead of committing.

The communication complexity of the protocol from [13] in our setting of parameters is poly$(k) \cdot s \cdot \log n$. We now describe how to reduce this complexity to $O(s) + \text{poly}(k)$ for the type of functionalities f required by our application.

The source of the multiplicative poly(k) overhead in [13] is the need to deal with circuits of an arbitrary depth. However, in the context of proof verification one may assume wlog that the witness is of size s (and includes the intermediate values of $C(w)$ in addition to w itself) and that R has s output bits, each verifying the consistency of one gate of C with its two inputs. (An output of 1^s of the augmented R is interpreted as an output of 1 of the original R; every other output of the augmented R is interpreted as 0.)

Note that each of the s output bits of R depends on only 3 of the s inputs bits. In this setting the total communication complexity of the protocol from [13] is only $O(s \log n) +$ poly(k, n). Here the $O(\log n)$ multiplicative overhead results from the secret sharing of [15], which requires the field size to be larger than the number of players. As a final optimization, one can implement the secret sharing of [15] over fields of a constant size using Algebraic-Geometric codes (analogously to their use in [9]). This results in an additional fractional decrease in the security threshold, which does not affect the asymptotic complexity of our construction.

Acknowledgments. We thank the anonymous STOC referees for helpful suggestions and comments.

5. REFERENCES

[1] O. Barkol and Y. Ishai. Secure Computation of Constant-Depth Circuits with Applications to Database Search Problems. In *Proc. Crypto 2005,* pages 395-411.

[2] M. Bellare, S. Micali, and R. Ostrovsky. The (True) Complexity of Statistical Zero Knowledge. In *Proc. of 22nd STOC*, pages 494-502, 1990.

[3] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. of 20th STOC*, pages 1–10, 1988.

[4] M. Blum. Coin Flipping by Telephone - A Protocol for Solving Impossible Problems. In *Proc. COMPCON 1982*: 133-137.

[5] J. Boyar, G. Brassard and R. Peralta. Subquadratic zero-knowledge. *J. ACM*, 42(6), pages 1169–1193, 1995. Earlier version in FOCS '91.

[6] J. Boyar, I. Damgård and R. Peralta. Short Non-interactive Cryptographic Proofs. *J. Cryptology* 13(4): 449-472 (2000).

[7] R. Canetti. Security and composition of multiparty cryptographic protocols. In *J. of Cryptology*, 13(1), 2000.

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

[9] H. Chen and R. Cramer. Algebraic Geometric Secret Sharing Schemes and Secure Multi-Party Computations over Small Fields. In *Proc. Crypto 2006.*

[10] R. Cramer and I. Damgård. Linear Zero-Knowledge - A Note on Efficient Zero-Knowledge Proofs and Arguments. In *Proc. STOC 1997*, pages 436-445.

[11] R. Cramer and I. Damgård. Zero-Knowledge Proofs for Finite Field Arithmetic; or: Can Zero-Knowledge be for Free? In *Proc. CRYPTO 1998*, pages 424-441.

[12] I. Damgård and Y. Ishai. Constant-Round Multiparty Computation Using a Black-Box Pseudorandom Generator. In *CRYPTO 2005*, pages 378–394.

[13] I. Damgård and Y. Ishai. Scalable Secure Multiparty Computation. In *Proc. CRYPTO 2006*, pages 501-520.

[14] S. Even, O. Goldreich and A. Lempel. A Randomized Protocol for Signing Contracts. In *Communications of the ACM,* 28(6):637–647, 1985.

[15] M. K. Franklin and M. Yung. Communication Complexity of Secure Computation. In *Proc. of STOC 1992*, pages 699-710.

[16] O. Goldreich. *Foundations of Cryptography: Basic Tools.* Cambridge University Press, 2001.

[17] O. Goldreich. *Foundations of Cryptography: Basic Applications.* Cambridge University Press, 2004.

[18] O. Goldreich and A. Kahan. How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. *J. Cryptology* 9(3): 167-190 (1996)

[19] S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM J. Comput.*, Vol. 18, No. 1, pp. 186-208, 1989.

[20] O. Goldreich and J. Håstad. On the Complexity of Interactive Proofs with Bounded Communication. *Inf. Process. Lett.* 67(4): 205-214, 1998.

[21] O. Goldreich, S. Micali, and A. Wigderson. How to Prove all NP-Statements in Zero-Knowledge, and a Methodology of Cryptographic Protocol Design. In *CRYPTO 1986*, pages 171-185.

[22] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game (extended abstract). In *Proc. of 19th STOC*, pages 218–229, 1987.

[23] J. Groth, R. Ostrovsky, and A. Sahai. Perfect Non-interactive Zero Knowledge for NP. In *Proc. EUROCRYPT 2006*, pages 339-358.

[24] I. Haitner and O. Reingold. Statistically-Hiding Commitment from Any One-Way Function. These proceedings.

[25] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A Pseudorandom Generator from any One-way Function. *SIAM J. Comput.* 28(4): 1364-1396 (1999).

[26] R. Impagliazzo and S. Rudich. Limits on the Provable Consequences of One-way Permutations. In *Proc. CRYPTO '88*, pages 8–26.

[27] Y. Ishai, E. Kushilevitz, Y. Lindell, and E. Petrank. Black-box constructions for secure computation. In *Proc. STOC 2006*, pages 99-108, 2006.

[28] Y. T. Kalai and R. Raz. Succinct Non-Interactive Zero-Knowledge Proofs with Preprocessing for LOGSNP. In *Proc. 47th FOCS*, pages 355–366, 2006.

[29] Y. T. Kalai and R. Raz. Interactive PCP. Manuscript, 2007.

[30] J. Kilian. Founding Cryptograph on Oblivious Transfer. In 20*th STOC*, pages 20–31, 1988.

[31] J. Kilian. A Note on Efficient Zero-Knowledge Proofs and Arguments (Extended Abstract). In *Proc. STOC 1992*, pages 723-732.

[32] J. Kilian and E. Petrank. An Efficient Noninteractive Zero-Knowledge Proof System for NP with General Assumptions. *J. Cryptology* 11(1), pages 1-27, 1998.

[33] J. Kilian, S. Micali, and R. Ostrovsky. Minimum Resource Zero-Knowledge Proofs. In *FOCS 1989*, pages 474-479.

[34] S. Micali. Computationally Sound Proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000.

[35] M. Naor. Bit commitment using pseudorandomness. *J. of Cryptology*, 4:151–158, 1991.

[36] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993. Preliminary version in Proc. STOC '90.

[37] M. Naor and K. Nissim. Communication preserving protocols for secure function evaluation. In *Proc. 33rd STOC*, pages 590–599, 2001.

[38] R. Ostrovsky and A. Wigderson. One-Way Fuctions are Essential for Non-Trivial Zero-Knowledge. In *Proc. of ISTCS 1993*, pages 3-17.

[39] M. Prabhakaran, A. Rosen, and A. Sahai. Concurrent Zero-Knowledge with Logarithmic Round Complexity. In *Proc. of FOCS 2002*, pages 366-375.

[40] M. Rabin. How to Exchange Secrets by Oblivious Transfer. Tech. Memo TR-81, Aiken Computation Laboratory, Harvard U., 1981.

[41] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proc. 21st STOC*, pages 73–85, 1989.

[42] A. Razborov. Lower bounds for the size of circuits of bounded depth with basis (AND, XOR). *Math. Notes of the Academy of Science of the USSR*, 41(4):333–338, 1987.

[43] O. Reingold, L. Trevisan, and S. P. Vadhan. Notions of Reducibility between Cryptographic Primitives. *Proc. 1st TCC*, pages 1-20, 2004.

[44] A. Rosen. A Note on Constant Round Zero Knowledge Proofs for NP. In *Proc. 1st TCC*, pages 191-202, 2004.

[45] R. Smolensky. Algebric methods in the theory of lower bound for boolean circuit complexity. In *Proc. 19h STOC*, pages 77–82, 1987.

[46] A. C. Yao. How to generate and exchange secrets. In *Proc. 27th FOCS*, pages 162–167, 1986.

APPENDIX

A. FULLY REALIZING ZK PROOFS

In the previous sections, many protocols were described in a hybrid model which assumes "idealized" versions of basic cryptographic primitives – in particular, commitment and coin-flipping. In this section, we sketch how well-known cryptographic techniques can be utilized to convert these protocols into zero-knowledge proofs in the plain model.

We stress that since our objective is to obtain *proofs* (not arguments), we cannot simply apply previous composition theorems to combine our protocols with composable commitment or coin-flipping, as these typically yield arguments and not proofs. Thus, we are forced to give specific details on what techniques we can use to maintain security against unbounded provers. Nevertheless, previously known techniques suffice for all our applications.

A.1 The Basic Protocol

We first examine the basic protocol from Section 3, which turns a *perfectly correct* and 2-private MPC protocol Π_f (in the semi-honest model) into a zero-knowledge proof protocol, Π_R, with constant soundness. Protocol Π_R follows the standard GMW zero-knowledge proof for 3-Colorability [21], and its analysis is essentially the same. We have already proven the soundness of the protocol. For the zero-knowledge property, we note that the verifier only has a polynomial number of choices it can make in the protocol. As such, the simulator works as follows:

1. The simulator makes the following kn^2 "attempts" until it succeeds. If the simulator fails kn^2 times, it aborts.
2. The simulator first chooses a pair of parties $\{i, j\}$ at random, and executes the MPC simulator to obtain the simulated views V_i and V_j of these parties. For all $h \notin \{i, j\}$, the simulator prepares random views V_h.
3. The simulator then prepares commitments to all the views above, and provides these to the verifier.
4. If the verifier responds with the challenge $\{i, j\}$, then we say that the "attempt" succeeds, otherwise the attempt fails, and we start over. If the attempt did succeed, then the simulator send openings to V_i and V_j to the verifier.

Clearly, each simulation attempt succeeds with independent probability at least $1/n^2$. Therefore the simulator succeeds at least once with probability $1 - 2^{\Omega(k)}$ by a Chernoff bound.

The computational indistinguishability of the simulation follows from a straightforward hybrid argument on the commitments, and by the computational indistinguishability of the MPC simulation.

A.2 Statistical Correctness

In the protocol of Section 3.1, we invoked an "idealized" coin-flipping protocol to choose a set of random strings. For use in our zero-knowledge proof, this coin-flipping protocol must enjoy the following properties:

1. (For Soundness:) With an infinitely powerful cheating prover, the distribution of random bits produced by the protocol must be statistically (negligibly) close to uniform.
2. (For Zero Knowledge:) There must exist a simulator, such that when given any randomly chosen string r, the simulator when interacting with any polynomial-time cheating verifier can force the output of the protocol to be either r or $\perp$, and such that the distribution of verifier views produced by the simulator (when given a uniformly random r as input) is computationally indistinguishable from an actual interaction between the honest prover and the cheating verifier.

We will outline an implementation of coin-flipping protocols with this property. Let m be the number of random bits to be chosen.

Sequential Blum. We consider the basic coin-flipping protocol of Blum [4] based on sequential repetition.
Repeat the following three steps m times:

1. The prover chooses a random bit r_P and provides a statistically binding commitment $\text{Commit}(r_P)$ to the verifier.
2. The verifier sends a random bit r_V to the prover.
3. The prover opens its commitment, and the output of this phase of the protocol is $r_P \oplus r_V$.

The soundness property follows from the statistical-binding property of the prover's commitment. The zero-knowledge property follows well-known arguments (see [4]). For self-containment, we sketch the simulation strategy and argument here: For each phase of the coin flipping protocol, on input a random bit r, the simulator makes k "attempts" (using independent randomness) to run the protocol acting as an honest prover, until it is able to force the output r. If it fails k times, the simulator aborts.

To show that this simulation succeeds, we need only argue that a dishonest polynomial-time verifier can only bias the output of each phase of the coin flipping protocol by a negligible amount. This follows by a simple reduction to the computational hiding property of the commitment scheme Commit. (The communication complexity of this coin-flipping is $O(mk)$.)

Application to zero-knowledge from statistically correct MPC. We turn to analyze the protocol from Section 3.1 combined with the above coin-flipping protocol. Because of the statistical soundness property of the coin flipping, the proof of soundness given earlier is not affected.

We now sketch the zero-knowledge proof and simulation. The simulation proceeds exactly as in the simulation of the basic protocol, with the following changes: (1) The simulation picks the $r_1^P, \ldots, r_n^P$ strings honestly, exactly as the honest prover would. (2) Then, it invokes the simulator of the coin-flipping protocol to force output strings $r_1^V, \ldots, r_n^V$ such that for the parties i, j, we have that $r_i^P \oplus r_i^V$ is equal to the random coins specified by the output of the MPC simulator. The analysis of zero-knowledge again follows from a straightforward hybrid argument.

A.3 ZK without Soundness Amplification

In the protocols of Section 4, where soundness amplification is done not by sequential repetition but by making use of a t-robust MPC protocol, to ensure the zero-knowledge property while preserving statistical soundness, we use ideas from [44, 39, 18, 2]. In particular, we add the following "preamble" in which the verifier commits in advance to its query $q = \{i_1, \ldots, i_t\}$ using a statistically-hiding commitment scheme:

1. The verifier commits to q and random strings $\{r_i^0\}_{i=1}^k$, $\{r_i^1\}_{i=1}^k$ of the appropriate length such that for all i, we have that $r_i^0 \oplus r_i^1 = q$ using the statistically-hiding commitment Com.
2. The prover sends k random bits $b_1 \ldots b_k$.
3. The verifier decommits to $\{r_i^{b_i}\}_{i=1}^k$.

Also, when the verifier later specifies its query q to the prover, the verifier must in addition open all of its commitments. The prover responds only if the openings are well-formed; otherwise, the protocol aborts.

The analysis of [44] shows that the simulator will learn the query q of the verifier (or learn if the verifier will cause the protocol to abort anyway) with overwhelming probability before the simulation moves beyond the "preamble". Therefore, the rest of the zero-knowledge simulation will proceed as before.

Note that the preamble above composes with the coin-flipping protocol given in the previous subsection. In the case of multi-phase protocols, such as our constant-rate ZK proof, the zero knowledge property is guaranteed because the simulator will learn the query $\{i_1, \ldots, i_t\}$ from the preamble simulation. Then the simulator can run the MPC simulator to obtain these views and the associated random coins. The coin-flipping simulator can be invoked to force the desired random coins as output, and the simulation proceeds as before.